A Simplified Fuzzy-Logic Control System Approach to Obstacle Avoidance combining Stereoscopic Vision and Sonar

by

Thomas Ian Grayston, BComp.

A dissertation submitted to the

School of Computing

in partial fulfilment of the requirements for the degree of

Bachelor of Computing with Honours

UNIVERSITY
OF TASMANIA

November 2006

Declaration

I hereby certify that this thesis contains no material which has been accepted for the award of any other degree or diploma in any tertiary institution, and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person except where due reference is made in the text of this thesis.

Signed: _____ Date: _____

Thomas Ian Grayston

Abstract

Stereoscopic vision is a technique for calculating the depths of objects in a scene from two images. Ultrasonic ranging is a well-established technique for estimating the distance to objects by bouncing an acoustic pulse off the object and measuring the time-of-flight. As with any types of sensors, these techniques each have their associated strengths and weaknesses. Therefore it is desirable to be able to use both sensor types simultaneously on a robot such that the benefits of the techniques can each be taken advantage of.

Effective obstacle avoidance is an important challenge in the field of robotics that is integral to achieving the goal of fully autonomous mobile robots. To achieve such behaviour a control system is required for directing the robot on the basis of sensor inputs.

This study presents a simplified fuzzy logic control system that differs from the standard fuzzy logic system in the way that fuzzy sets are generated. The control layers of the system dynamically create fuzzy sets on-the-fly when called upon to do so. The developed control system is used to show that there are benefits to combining stereoscopic vision and sonar for robot obstacle avoidance compared against using these sensors in isolation.

Acknowledgements

Firstly, thanks go to my supervisor, Dr Daniel Rolf, for his guidance, suggestions and reassurances throughout the year.

Many thanks go out to various friends who have been of great assistance. Thankyou to Eloise for assistance with taking measurements, and to Stuart for proofreading drafts and making many suggestions for improvement. Thankyou also to Emily for advice in regards to writing correct English. I would also like to thank Humphrey for his technical assistance in using LaTeX. Thanks to Kylie for putting up with the many long days and nights spent completing this thesis.

Thanks go to the School of Computing technical staff: Andrew, Rick, and Christian. They are always willing to provide assistance beyond the call of duty. Thanks to Tony for the purchasing of equipment.

Table of Contents

List of Figures

Chapter 1

Introduction

One of the primary goals of the field of robotics is to develop machines that can operate independently, achieving complex tasks without human supervision. A scenario could be imagined where robots exist alongside the human population, carrying out tasks that assist humans in their day-to-day tasks. Such a scenario is the focus of many works of science fiction in literature. Developing methods for autonomous robot navigation is essential for the realisation of this dream. An important aspect of autonomous robot navigation is obstacle avoidance. The hypothesis under examination, then, is:

> that combining short range sonar and stereoscopic vision using a simplified fuzzy logic control system is better than using these sensors in isolation in enabling the flexible robot platform to avoid obstacles.

For the purposes of this study, "better" obstacle avoidance is defined as detecting an obstacle as early as possible to allow manoeuvring around that obstacle. Consider the case in which a robot must traverse a field of landmines. It would be wise to allow as much clearance as possible rather than driving as close to the 'danger zone' as possible. Thus, early detection and the ultimate avoidance of obstacles are the keys to success. Whilst it might be considered advantageous to also examine what occurs after the robot has cleared the obstacle, this is outside the scope of this study, and will be left to further work.

There are many sensor types available for use in robotics. Each type of sensor, and for that matter each specific sensor itself has a wide range of characteristics and operational parameters. These may include range, granularity, or type of material sensed, just to name a few. Combining sensor inputs in such a way as to exploit the advantages of each sensor type is desirable.

The focus of this study, therefore, is on the control system. The study is restricted to the use of sonar and vision in order to keep the scope of the study manageable.

1.1 Thesis structure

First a review of the literature covering topics involved in this thesis will be discussed in Chapter 2. This will include the topics of stereoscopic vision, sonar and robot control system architectures.

Chapter 3 will then discuss the methodology used to test the hypothesis. The method used to calculate the depth of an obstacle using a stereoscopic vision system and sonar sensors will be covered. The control system that was devised for this study will be described, including the additional goal that was balanced with obstacle avoidance in the form of a compass bearing. A discussion of the various sources of error present in the sensors and measurement apparatus will also be discussed.

The results obtained are presented in Chapter 4 along with a discussion of those results. This includes measurements of the error present in the stereo vision system and the digital compass.

Finally, conclusions are drawn in Chapter 5 along with suggestions for further work that could extend the work completed by this study.

Chapter 2

Literature Review

2.1 Introduction

The literature that was reviewed in relation to this study covers a number of fields. Stereoscopic vision techniques are discussed as well as software technologies relating to computer vision. This is followed by a discussion of sonar technology and the problems associated with it. Finally, control systems are discussed, including an overview of the deliberative paradigm, the reactive paradigm, subsumption, and fuzzy logic control systems.

2.2 Stereoscopic Vision

Stereo is a broad term indicating the involvement of three-dimensional space (Nalwa 1993). In the case of computer vision it typically refers to the extraction of three-dimensional information from multiple images of a scene. The first robot to successfully use stereo vision was Moravec's (1981) Stanford Cart (Murphy 2000). It is possible to involve any number of points-of-view in stereo vision, captured by a single moving camera or a number of stationary ones. Three-dimensional information about a scene can be calculated by using a fairly simple geometric model of the camera positions and their field of projection (Nalwa 1993). The depth of this research, however, will be limited to the use of a pair of images taken by two cameras simultaneously (or effectively simultaneously).

2.2.1 Overview

In order to extract stereo information, it is necessary to identify matching points on each image that correspond to the same point in real space. Once this is

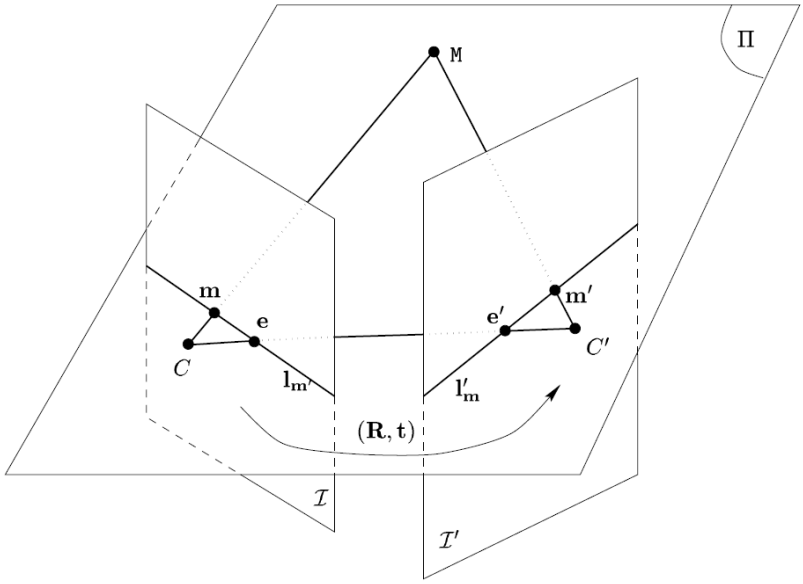


Figure 2.1: The geometry involved in geometric stereo vision (Zhang 1998)

done, triangulation can be used to calculate the position of the point in three-dimensional space. Figure 2.1, taken from Zhang (1998), shows the geometry involved. M is a point in real space that exists as part of an object in the scene. This is known as the *object point*. C and C' are the centres of projection for the left and right cameras respectively. I and I' are the image planes that the left and right camera views are projected onto. m is the point on the 2D image plane I at which the object point is observed. This point is referred to as an *image point*. The challenge of stereo correlation, then, is to identify *conjugate image points*; that is, pairs of image points, one from each image plane, that correspond to the same object point (Nalwa 1993).

2.2.2 Epipolar Geometry

Having identified a point of interest on one image plane, the next step is to identify a matching point on the second image plane. A property of the geometry called the *epipolar constraint* means that it is not necessary to search the entire second image for this point of interest.

The line joining the two centres of projection (CC') is called the *baseline*. Any plane through the baseline is known as an *epipolar plane* (II). The straight line that occurs at the intersection of an epipolar plane and an image plane is termed an *epipolar line*. Figure 2.1 contains two epipolar lines: $l_{m'}$ and l'_m.

Thus, the corresponding image point of m is constrained to lie on the epipolar line l'_m. Similarly, the corresponding image point of m' is constrained to lie on its epipolar line $l_{m'}$. This is of great benefit, as the problem of finding a match for a given point is reduced from a 2D problem to a 1D one. This results in a reduction in computation requirements as well as increasing the likelihood of obtaining correct matches (Nalwa 1993, Zhang 1998).

2.2.3 Calibration

There are two major categories of camera calibration parameters that apply to vision systems. These are intrinsic parameters and extrinsic parameters (Tsai 1987). Intrinsic parameters deal with the optical properties of the camera, such as lens distortion etc. Extrinsic parameters deal, essentially, with the precise positioning of the cameras with respect to each other.

In order to make depth calculations from a stereo pair it is important that the image planes be coplanar and parallel to their baseline. Such images are said to be rectified (Nalwa 1993). This can be accomplished by carefully positioning the cameras physically, or by applying transformations to the images after they are captured (Robert et al. 1994). It is possible for these transformations to be calculated somewhat automatically.

Typical camera calibration techniques involve the use of a known calibration object. This object typically features a carefully devised pattern of markings that allow the system to determine the intrinsic and/or extrinsic parameters of the setup. It has also been demonstrated that it is possible to determine the necessary transformations without the use of a calibration object; the scene itself can be used for calibration (Faugeras et al. 1992, Zhang et al. 1995, Nalwa 1993). Calibration is commonly applied at the beginning of a set of experiments, however calibration can be lost during the execution of a task (Zhang & Schenk 1997), severely impacting the effectiveness of the stereo vision system. Zhang & Schenk (1997) and Deriche et al. (1994) present methods for maintaining camera calibration during execution of tasks.

2.2.4 Correspondence Establishment

Once a pair of rectified stereo images have been established, the main task of identifying conjugate image points can be considered.

Intensity-Based Methods

A simple approach to establish correspondence between a pair of images is to compare the intensity of pixels (Nalwa 1993). This assumes that conjugate

image points do indeed have matching intensities. As discussed, the search can be limited to only occur along conjugate epipolar lines. It is likely that several points along epipolar lines will have closely matching intensities, however, so matching on a point-by-point basis will not be sufficient. Instead, matching can occur on a region-by-region basis. The size of the regions must be selected carefully. If they are too small, correct matches may be missed. If they are too large, localised variations in depth within individual regions may be too great.

One enhancement to the basic intensity based method involves the use of multiple resolutions of the image pairs. A reduction in the resolution of an image pair increases the likelihood that they will match, and allows smaller search regions to be used. Once the disparity at low resolution has been established, increasingly higher resolutions can be considered, using the disparity of the previous lower resolution as a guide for calculating the more detailed disparity (Iocchi & Konolige 1998). A further variation employs stochastic relaxation at each resolution to refine the matching of the previous resolution (Nalwa 1993).

Edge-Based Methods

Edges in images can be localised fairly accurately, and so are well suited for use in geometric stereo matching. Edge-based methods first perform edge-detection on individual images from a stereo pair, then attempt to match edges between the two images, again searching along conjugate epipolar lines. This approach requires that edges exist in the scene, and that edges are viewpoint-independent. The perceived 'edges' of an upright cylinder will appear to be in different locations depending on the viewpoint. These limitations limit the usefulness of edge-based methods in their own right, but can be particularly useful when used in combination with intensity-based correlation methods (Nalwa 1993).

Binary image thresholding is a simple technique for identifying objects in a scene based on colour (Murphy 2000, p.226). For example, robot programmed to collect red Coca-Cola cans might only consider pixels in the image that are what it considers to be 'red'.

2.2.5 Machine Vision Software Technologies

There are a number of software technologies available that provide image capturing and analysis capabilities. This section details the major options available that might be suited to the stereoscopic vision problem in the context of mobile robotics. The main requirements for the vision technology to be used in this study are:

- Ability to capture from usb webcam sources

- Ability to capture from two sources simultaneously or at least in sequence
- Compatibility with the existing code written in Java
- Cost-effectiveness (preferably free)

Java Media Framework

The Java Media Framework (JMF) is a Java API developed by Sun Microsystems that allows audio, video and other time-based media to be used in Java applications and applets (Sun Microsystems, Inc. 2006). JMF employs the use of Video For Windows (VFW), a legacy media framework developed by Microsoft which has a number of known deficiencies. VFW only defines a user-mode interface, meaning that many options can only be selected via user input, including video source selection. In other words, there is no way to switch between video sources programmatically.

VFW should, however, allow multiple video sources to be used simultaneously, once selected by the user. Unfortunately JMF does not fully support this. There are a number of threads on Java Technology Forums discussing this issue. In response to a question about using multiple webcams with JMF, one user writes: "I have two webcams that JMF can detect and work with separately but when connected at the same time JMF only recognizes one of them." (Mikeklein34 2004).

Third party image capture applications

One option that was considered was the use of a third party application for capturing the images. The ideal application would be able to capture images from two usb webcams simultaneously, or at least in sequence programmatically. It would save the two images to two files where another program (possibly written in Java) could open them and perform the stereo analysis. Such an application would either run continuously in the background capturing images at regular intervals, or it would capture a single image (or pair of images) on launch and then terminate. It would also be low-cost, preferably free.

Whilst there were many applications that fulfilled some of these requirements, no one program had all the features required. It proved particularly difficult to access two cameras at the same time.

Open Source Computer Vision Library

The Open Source Computer Vision Library (OpenCV) is a library of computer vision functions developed by Intel. It is primarily aimed at real-time computer

vision problems such as those found in human-computer interface, robotics, monitoring, biometrics and security (Intel Corporation 2006).

DirectShow

"The Microsoft DirectShow application programming interface (API) is a media-streaming architecture for the Microsoft Windows platform. Using DirectShow, your applications can perform high-quality video and audio playback or capture." (Microsoft Corporation 2006)

Microsoft developed DirectShow as a replacement for VFW. It was named DirectShow in 1998 to align the technology with the DirectX family, though it was moved from the DirectX SDK to the Microsoft Platform SDK in April 2005. It was then removed from the Platform SDK in 2006. DirectShow is capable of multiple simultaneous video streams.

C#

Microsoft Visual C# is a language from Microsoft's .NET framework. It is a popular language for application development for Microsoft Windows. It is well-suited to vision applications as the programming environment is very visually-oriented, providing built-in functionality for loading and saving images to/from file, for example. Built-in libraries include things such as the Color class which provides useful functionality such as RGB to HSV colour-space conversion. It is designed with building rapid prototypes in mind which means it is easy to build applications quickly. It is also suitable for use with DirectShow.

2.3 Sonar

In the context of mobile robots, the term "sonar" refers to airborne ultrasonic range sensing, also known as acoustic ranging. Whilst it is possible to use frequency modulation, Doppler, and other techniques, time-of-flight has thus far been the dominant method used in sonar (Leonard & Durrant-Whyte 1992). Acoustic ranging by time-of-flight is the technique of judging distance by bouncing an acoustic pulse off an object, measuring the time between the pulse and the reflected echo (McComb 2000). Figure 2.2 (Hall 2005) illustrates a sonar device and the principle behind time-of-flight. This research was focussed on indoor applications, but sonar has been shown to function well in outdoor environments as well (Langer & Thorpe 1997).

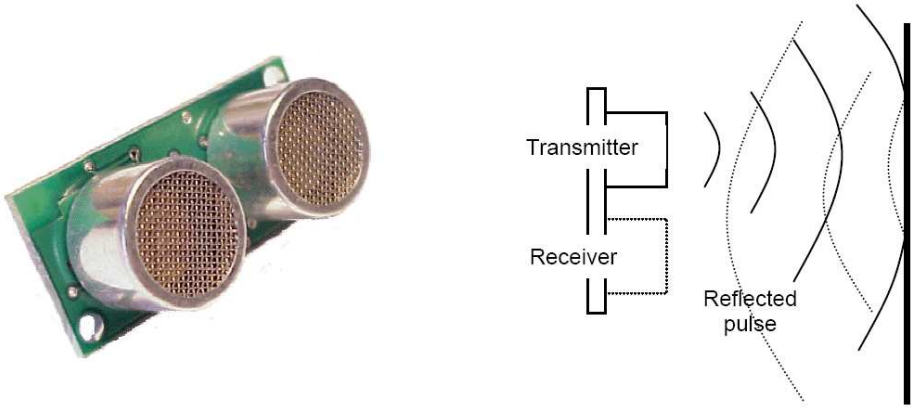


Figure 2.2: SRF04 Ultrasonic rangefinder module and diagram of operation (Hall 2005)

2.3.1 Problems associated with Sonar

Sonar is a very useful sensor type that has received a great deal of attention in the field of robotics. There are, however, a number of problems associated with sonar that require consideration (Leonard & Durrant-Whyte 1992).

Imprecision Since the sonar beam is cone-shaped, the precise position of any detected obstacle cannot be determined from a single reading. The object could exist anywhere on the arc described by the cone (Nehmzow 2003).

Specular Reflections A sonar beam hitting a smooth surface at a shallow angle is reflected off the surface away from the robot, and is therefore not detected. The robot only receives a response if there is an object further away that reflects the beam back. Either way, the robot will interpret the reading to mean that there is more free space than actually exists (Nehmzow 2003).

Crosstalk Where more than one sonar sensor is used, one sensor may detect the beam that was produced by another, again giving false readings (Nehmzow 2003).

There are techniques for minimising the effects of these problems. Delgado et al. (1998) "propose[s] a sensor validation method based on history" to attempt to reduce the affects of "well known problems like crosstalking, noise, and poor reliability for incidence angles bigger than 15 degrees in conjunction with smooth surfaces."

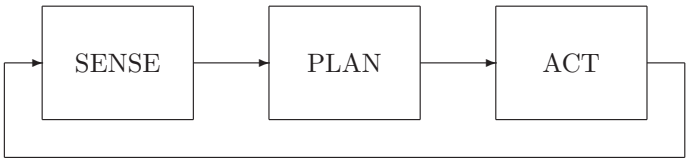


Figure 2.3: SENSE-PLAN-ACT organisation of the Deliberative Paradigm (Murphy 2000)

2.4 Control System Architecture

Gathering information about the environment from various sensors is the first step in achieving autonomous robotic navigation, but determining what to do with the information is critical to success. The problem of determining what action to pursue at any given moment is not a trivial one. **** The system must be able to consider the various sensor inputs and the overall goals that have been set and combine these in such a way as to produce a specific action, such as moving forwards or turning to the left or right. In the field of robotics, there have been three major approaches to control system architecture over the last four or five decades: The deliberative (or hierarchical) paradigm, the reactive paradigm, and hybrid approaches (Murphy 2000).

2.4.1 The Deliberative Paradigm

The Deliberative Paradigm (also known as the Hierarchical Paradigm) was the first widely-used approach to the problem of organising intelligence in mainstream robotics (Murphy 2000). It dominated robot control system design from 1967 until it was overtaken by the Reactive Paradigm (discussed in the next section) in the late 1980's. As shown in Figure 2.3 (Murphy 2000), the deliberative paradigm divides the problem of robotic control into three stages: SENSE, PLAN, ACT. These stages cycle repeatedly while the robot is functioning.

The SENSE stage takes input from sensors, building and maintaining a data structure called the world model. The world model contains all the information about the environment that the robot is aware of. The PLAN stage uses data from the world model to construct a list of directives that will accomplish the robot's goal. The ACT stage then attempts to execute only the first directive in this list. After completing a SENSE-PLAN-ACT sequence, the procedure is repeated, sensing how the environment has changed from the robot's actions, making a new plan, and carrying out the first directive specified by the new plan (Murphy 2000). A mobile robot called SHAKEY, designed by the Stanford Research Institute in the late 1960s, was arguably the first mobile robot to

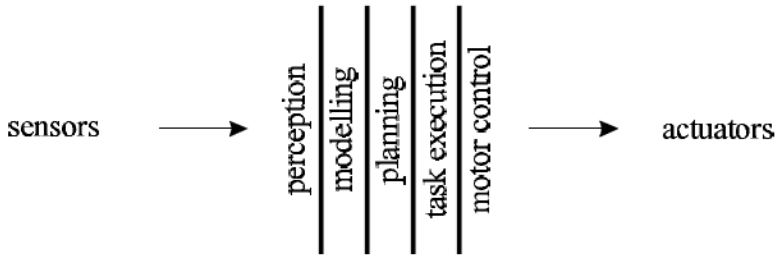


Figure 2.4: The Classical Artificial Intelligence Approach to Robot Control (Flanagan et al. 1995)

employ the SENSE-PLAN-ACT approach (Flanagan et al. 2003).

In a typical implementation of a system that falls under the deliberative SENSE-PLAN-ACT paradigm, the steps are implemented as one or more subsystems that each deal with a specific stage of the problem. Figure 2.4 (Flanagan et al. 1995) shows a typical set of subsystems: perception, world modelling, planning, task execution and motor control. These subsystems can be thought of as horizontally arranged "vertical" slices of the overall problem, each being a complex component that requires a particular type of input and produces a particular type of output. Because each subsystem deals with a well-defined part of the problem and is unable to control the robot on its own, each slice must work correctly for the robot to function correctly. This impedes development, as any significant changes that are made to one slice will affect the others also (Flanagan et al. 1995).

The planning phase represents a significant bottleneck within the deliberative paradigm (Murphy 2000). The planning algorithms employed are typically extremely slow, which results in a robot that does not react very quickly to changes in its environment. Planning that is being undertaken is being conducted on a model that may be significantly out-of-date. Because the sense and act phases are completely decoupled, the robot is unable to exhibit any stimulus-response types of actions. In some situations there may not be enough time to make the best decision possible. For example, in the case of a rock falling towards the robot, the robot would be wise to quickly move *anywhere* to avoid being crushed, rather than stopping to determine the optimal direction to travel in. A deliberative system employs a closed world assumption, which means that the model is assumed to be accurate and up-to-date, and this is certainly not always the case.

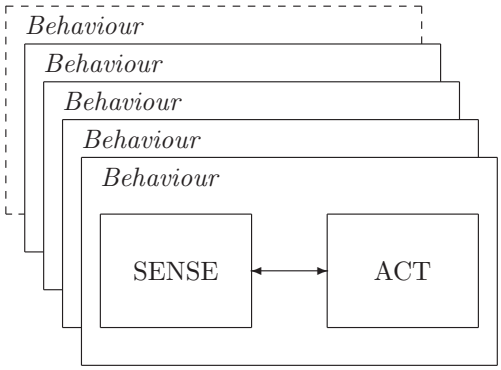


Figure 2.5: SENSE-ACT organisation of the Reactive Paradigm into multiple, concurrent behaviours (Murphy 2000)

2.4.2 The Reactive Paradigm

Emerging in the late 1980's, the Reactive Paradigm attempts to solve some of the problems of the Deliberative Paradigm and takes much of its inspiration from observations of intelligence in biological systems (Murphy 2000). As shown in Figure 2.5 (Murphy 2000), it completely discards the PLAN component of the SENSE-PLAN-ACT sequence, instead utilising tightly-coupled SENSE-ACT pairs arranged in multiple behaviours. This approach is in line with Brooks (1991), who argues that research in the field of intelligent systems must focus on the development of simpler systems that initially aim for something much less complex than human-level intelligence. Such systems should be built up incrementally, such that a fully working system exists at each stage. He also argues that robots should react directly to their sensed environment, rather than using an intermediate world model. This avoids the representation and currency problems associated with the closed world assumption.

Subsumption

An alternative approach to the classical, deliberative, horizontally arranged subsystem approach as discussed in the previous section, is an approach called Subsumption. Developed by Brooks (1986), Subsumption is a purely Reactive system that involves multiple SENSE-ACT behaviours arranged in a vertical hierarchy of layers. Each layer provides a simple behaviour in its entirety, rather than an incomplete part of an overall complex behaviour. An example set of behaviours is illustrated in Figure 2.6 (Flanagan et al. 1995). Under Brooks' (1986) original Subsumption architecture, each layer subsumes the one under it, adding behaviour that builds on the previous one. The highest layer has the highest priority.

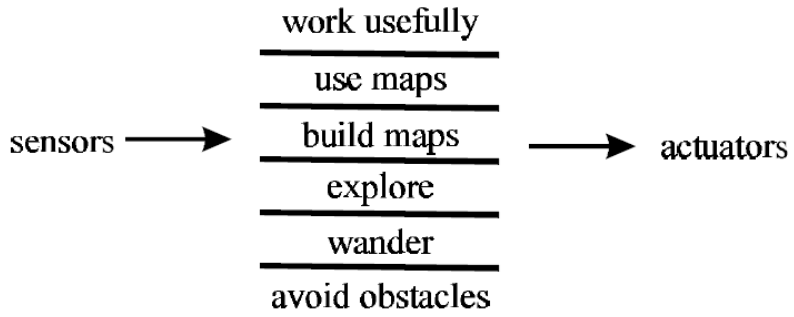


Figure 2.6: Subsumption Architecture (Flanagan et al. 1995)

Flanagan et al. (2003) argues that this priority ordering results in layers that are too dependent on the layers beneath them. Each layer must be engineered to not only add the new behaviour, but to be responsible for incorporating all previous behaviours as well. This considerably adds to the complexity of designing new layers once the system has more than a few behaviours. One of Subsumption's original primary aims was to enable systems to be built that consist of behaviours that operate independently of one another, enabling behaviours to be added and subtracted without any other modifications. This is not possible under Brooks' (1986) approach, as each layer is tightly coupled with the layers below.

Flanagan et al. (2003) addresses this problem by presenting a modified approach called *Enhanced Subsumption Control*. Under this approach, the ordering of priorities is reversed—the lowest-level behaviour (e.g. avoid obstacles) is assigned the highest priority. The justification for this change is that it is the lowest levels of the hierarchy that must react most swiftly to a changing environment, whereas higher layers deal with longer-term goals. In this way, the layers are still arranged in a hierarchy, but each behaviour exists in isolation to the others. Each layer is individually capable of fully controlling the robot, even though the behaviour produced by each layer in isolation may be very basic.

For example, one layer might direct the robot towards a target. The lower "avoid obstacles" layer will not assume control until it detects an obstacle. When this happens it assumes control, directing the robot away from the obstacle until the robot is clear. Thus the robot will head towards the target location, avoiding obstacles along the way.

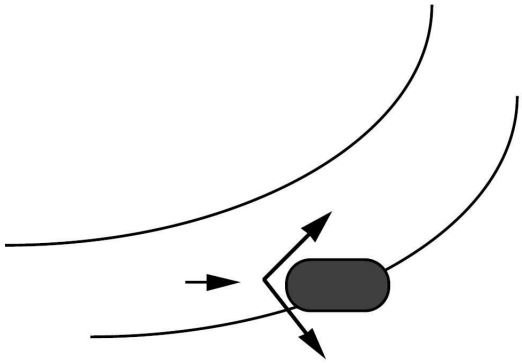


Figure 2.7: Which way to turn? Obstacle avoidance does not know (Yen & Pfluger 1995)

Limitations of Subsumption

One limitation of the standard Subsumption architecture is that when a particular layer has control, it has no knowledge of the goals of other behaviours (Leyden et al. 1999, Toal et al. 1996). This can result in erratic behaviour under certain circumstances. For example, consider a system with an avoid-obstacles layer as well as a follow-path layer. As illustrated in Figure 2.7 (Yen & Pfluger 1995), the avoid-obstacles behaviour must decide which way to turn to avoid a collision. Since it has no knowledge of follow-path's preferred direction, it might arbitrarily choose to turn to the right, taking the robot away from its higher-level objective of following the path.

A related problem that may occur under Subsumption is rapid switching back and forth between layers when conditions are borderline for a particular behaviour (Yen & Pfluger 1995). Using the example above, upon sensing the obstacle the avoid-obstacles layer takes control and begins turning the robot away. The moment the obstacle is out of range once again, control is returned to the follow-path behaviour. If the avoid-obstacles layer turned the robot to the right, the follow-path layer turns the robot back towards the obstacle. This results in the obstacle again being sensed as an obstruction. The avoid-obstacles layer again assumes control. This switching back and forth between layers is continued until the robot is clear of the obstacle. The resulting motion resembles a 'zigzag' pattern.

Extensions of Subsumption

Toal et al. (1996) suggests the following variations to the Subsumption architecture. Rather than arranging behaviours in a hierarchy, have them exist in

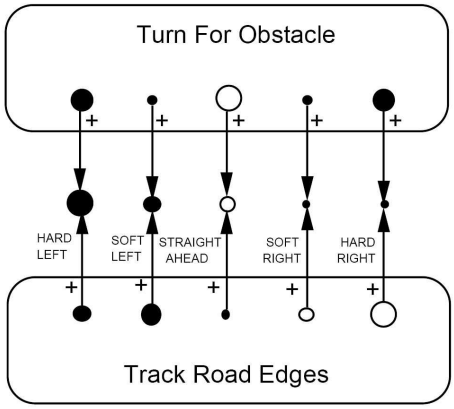


Figure 2.8: A Payton-Rosenblatt network for fusing travel direction commands of two behaviours (Yen & Pfluger 1995)

an unordered set, using a separate scheduler to determine which layers are allocated control at which times. A shared global data structure called the 'blackboard' is used to share information between behaviours. Again considering the avoid-obstacles/follow-path example, this approach allows the avoid-obstacles behaviour to read information off the blackboard to determine the left or right turning preferences of other behaviours in the system.

Rosenblatt & Thorpe (1995) present a similar approach, employing a central arbiter that "receives votes for and against commands from each subsystem and decides upon the course of action which best satisfies the current goals and constraints of the system."

The core limitation under Subsumption is that layers are completely unaware of the goals of other layers, and so non-active layers can have no influence over the sometimes arbitrary decisions of higher-level layers (Payton et al. 1990). Payton et al. (1990) presents a system in which each layer outputs preferences for each of the possible control commands. These command preferences are then fused together to produce the final output that is executed. This command fusion is accomplished using the so-called *Payton-Rosenblatt command fusion network* as shown in Figure 2.8.

In the pictured example, there are two behaviours contributing to the decision on turning direction: Turn-for-Obstacle and Track-Road-Edges. The five possible outcomes are hard-left, soft-left, straight-ahead, soft-right, and hard-right. Filled circles represent a positive weight and unfilled ones represent a negative weight. The diameter of each circle represents the level of activation for that unit. As can be seen from the diagram, Turn-for-Obstacle has a definite preference against steering straight-ahead, indicating that this would result in a

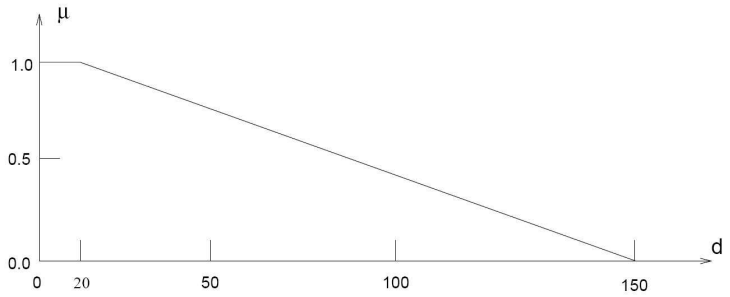


Figure 2.9: A Fuzzy Set representing the concept of Near (Yen & Pfluger 1995)

collision. It therefore most highly favours turning hard-left or hard-right. The Track-Road-Edges behaviour favours turns that keep it centred on the road. In this instance it favours turning soft-left most highly, with the values dropping away gradually from that option through to the strong negative weighting against turning hard-right.

The final command units (middle) take their values from a weighted sum of the corresponding behaviour units. Note that the Turn-for-Obstacle behaviour is assigned a higher priority than the Track-Road-Edges behaviour, so its outputs weigh more heavily into the final output. In the example, the resulting command unit with the highest activation level is hard-left, so this is the final output that is selected (Payton et al. 1990).

2.4.3 Fuzzy Logic Control System

Fuzzy Logic is essentially a system for dealing with uncertainty and vagueness-of-concept (Lewis 1997). The system was first proposed by Zadeh (1965) and there are now many variations on the concept of a Fuzzy Logic system. Fuzzy Logic allows objects to take partial membership in vague concepts. It achieves this through the use of a structure called a *fuzzy set*. An example of a fuzzy set is shown in Figure 2.9 (Yen & Pfluger 1995), showing a representation of the concept of Near. Nearness is a property that has no specific cutoff; there's no particular point at which an object transitions from being near to not being near. In this example, an object that is within 20 units has full membership in Near. At 100 units away it has 50% membership, and at 150 units or more an object is no longer Near at all. Using representations of concepts that facilitate vagueness in robot control systems is desirable as it can make these systems more robust (Yen & Pfluger 1995). Errors such as noise will only affect a value's membership to a fuzzy set by a small degree, which will result in only

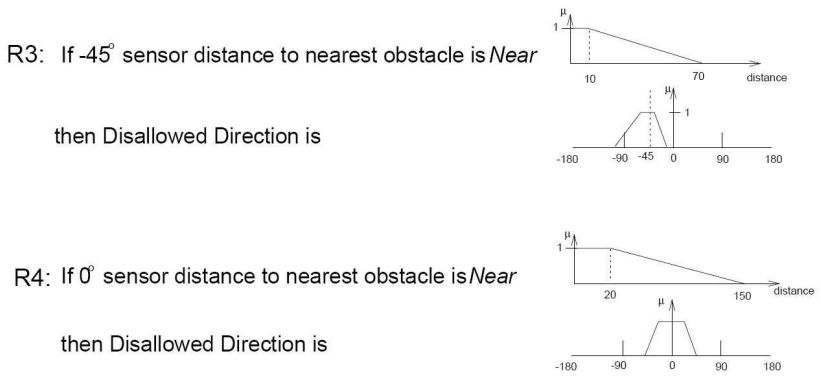


Figure 2.10: Example fuzzy rules used for obstacle avoidance (Yen & Pfluger 1995)

minor changes to final control commands.

At its core, a fuzzy logic control system consists of a set of control rules contained in a rule-base. Each rule has one or more antecedents and one consequent which are represented using fixed fuzzy sets. Figure 2.10 (Yen & Pfluger 1995) shows two example rules used for obstacle avoidance. These two rules take input from different sensors and apply output to a different fuzzy set, but both contribute to the Disallowed Direction variable. The distance returned by the two sensors in question will determine the result that is returned by the Near fuzzy set. This value (between 0 and 1) is used to scale the output fuzzy set. Each of the rules associate their fuzzy set output with a particular fuzzy conclusion (e.g. Disallowed Direction). Each fuzzy conclusion then has a number of fuzzy sets associated with it. These are combined together into a single composite fuzzy set, typically through the use of a logical AND operation. An example of this is shown in Figure 2.11. In this case there are a number of sensors contributing to the resultant fuzzy set. This is an example of sensor fusion.

The next step is to combine the fuzzy conclusions together. In the example shown in Figure 2.12, there are two fuzzy conclusions: *Desired Direction* and *Allowed Direction* (where *Allowed Direction* $= 1-$ *Disallowed Direction*). Each of these represents a contradictory command to the robot telling it to turn in a particular direction. Because they are represented as fuzzy sets, however, it is possible to fuse the commands together, producing a final *Turning Direction* fuzzy set. In this way the desires of each behaviour are preserved, incurring no information loss.

The last step is to use this final fuzzy set to determine the optimal turning direction in the form of a 'crisp' output—a single real number value, as opposed to a fuzzy set. This process is called defuzzification, and there are three major

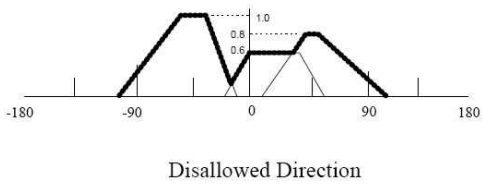


Figure 2.11: An example sensor fusion and the resultant fuzzy set (Yen & Pfluger 1995)

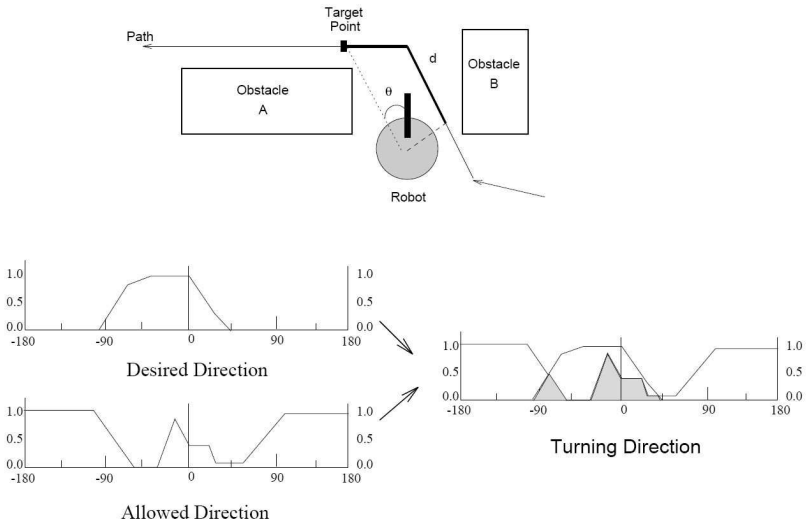


Figure 2.12: An example command fusion (Yen & Pfluger 1995)

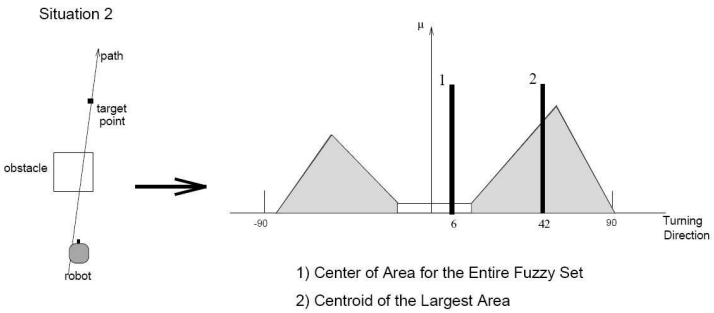


Figure 2.13: An example of defuzzification (Yen & Pfluger 1995)

techniques for doing this: Mean of Maximum, Centre of Area, and Centroid of the Largest Area (Yen & Pfluger 1995). The Mean of Maximum (MOM) method returns the average of those values sharing the maximum membership value for that set. The Centre of Area (COA) method returns the centre of gravity of the entire fuzzy command. The Centroid of Largest Area (CLA) method (developed by Yen & Pfluger (1995)) first divides a multiple-peak fuzzy command into several fuzzy subsets. COA is then applied to the subset with the largest area. The COA and CLA methods are shown in an example in Figure 2.13. As shown, COA selects a direction that sits between the two peaks in this example. This is very undesirable in the case of obstacle avoidance—rather than selecting just one of the two options (left or right) for avoiding the obstacle, the robot chooses to "compromise" between the two options and drives forwards, most likely resulting in a collision. The MOM approach will select a desirable outcome provided there is only one local maximum. If the left and right peaks are equal, however, it will respond in much the same way as COA. A major disadvantage of MOM is that it does not use all of the information contained in the fuzzy command, which results in difficulty in producing behaviour that changes smoothly over time. The CLA method is the most popular. In this case it selects the fuzzy subset on the right as it has the largest area, and returns a result guiding the robot away from the obstacle. It is able to produce behaviour that changes smoothly over time because of its use of centre of gravity.

Yen & Pfluger (1995) notes that Payton-Rosenblatt's (P-R) approach, whilst not described in terms of a fuzzy approach, bears much similarity to fuzzy systems. The activation levels of units in the P-R network correspond to membership degrees in fuzzy sets. Each behaviour, as represented by a P-R network (as in Figure 2.8) is analogous to a discrete fuzzy set. Thirdly, the method for selecting of the final output (the maximum activation level) is essentially that

of the MOM defuzzification method.

2.5 Conclusion

This chapter has covered key topics in the literature that relate to this study. Various stereoscopic vision techniques were discussed. An overview of the options for computer vision software was covered. Sonar was introduced along with its sources of error. The major developments in robotic control systems were covered, beginning with the deliberative paradigm. A major change in control systems thinking occurred with the introduction of the reactive paradigm. The epitome of the reactive paradigm was the subsumption architecture which grew into the type of system now known as the fuzzy logic control system.

Chapter 3

Methodology

This chapter presents the methodology employed in testing the hypothesis of this study. Specific technologies were chosen for use in the implementation of software that was created in order to conduct the experiments. The physical environment was constructed in such a way as to standardise tests and take measurements as accurately as possible.

3.1 Stereoscopic Vision

The stereoscopic vision system was developed as a separate component to the rest of the robot control system software. The reasons for this, and details on how the system is implemented are presented in this section.

3.1.1 Machine Vision Software Technologies

It was initially hoped that the vision system could be implemented as part of the existing flexible robot platform software which was written in Java. As discussed in Chapter 2, the Java Media Framework (JMF) uses Video for Windows (VFW) for capturing images, which does not allow multiple simultaneous video capture streams, nor the programmatic selection of video sources. For this reason JMF was rejected as an option.

An alternative was to use a third-party application for capturing images to file, then reading these images into the Java program. This approach was also abandoned after evaluating a variety of programs and determining that none of them could meet all of the requirements.

At this point it was determined that some or all of the stereo vision component must be implemented in a language other than Java. OpenCV (see Chapter 2) was considered for its powerful image processing library, but the amount

of time required to become familiar with it would have been prohibitive. In any case, most of the available features are more sophisticated than would be useful for this study.

Microsoft DirectShow is a current API that can be used for streaming video under Microsoft Windows. C# is a popular language that can easily employ the use of DirectShow. These two technologies, DirectShow and C#, were selected for developing the Stereo Vision system.

3.1.2 Software Architecture

Whilst it is theoretically possible to capture images from two video sources simultaneously using DirectShow, the implementation of such a system is complex. For the purpose of the experiments conducted, it was a requirement only that the two images be obtained without user interaction. It was considered acceptable that the robot remain in a fixed position during the image capture process. In order to simplify software development, the system first takes an image from the left camera, then from the right camera.

The Stereo Vision component itself is divided into two parts: The StereoGrabber component and the StereoAnalysis component.

The StereoGrabber component is responsible for capturing an image from a webcam source. It takes a parameter specifying which camera to use, initialises the appropriate camera and captures a still image. The code used to do this is based on source code from Audrey Mbogho (2005). The captured image is then saved to a file and flow-of-control is returned to the calling application.

The StereoAnalysis component analyses a stereo pair in order to determine the depth of features in the environment. It does this by using the StereoGrabber component to capture the two images, then making depth calculations, writing the results to a file. Flow-of-control is then returned to the calling application. The method used for calculating the depth of features from a stereoscopic pair is discussed next.

3.1.3 Identifying image features

Before depth can be calculated, the object points in the scene on which the depths are to be calculated must be established. The focus of this research was to establish whether it is beneficial to combine stereoscopic vision and sonar. It was not concerned with the quality of the stereoscopic vision technique itself. It was therefore not necessary to create a particularly robust or accurate stereo vision system. The requirements for the stereo vision system, then, were that it be able to consistently provide sufficiently accurate depth estimations

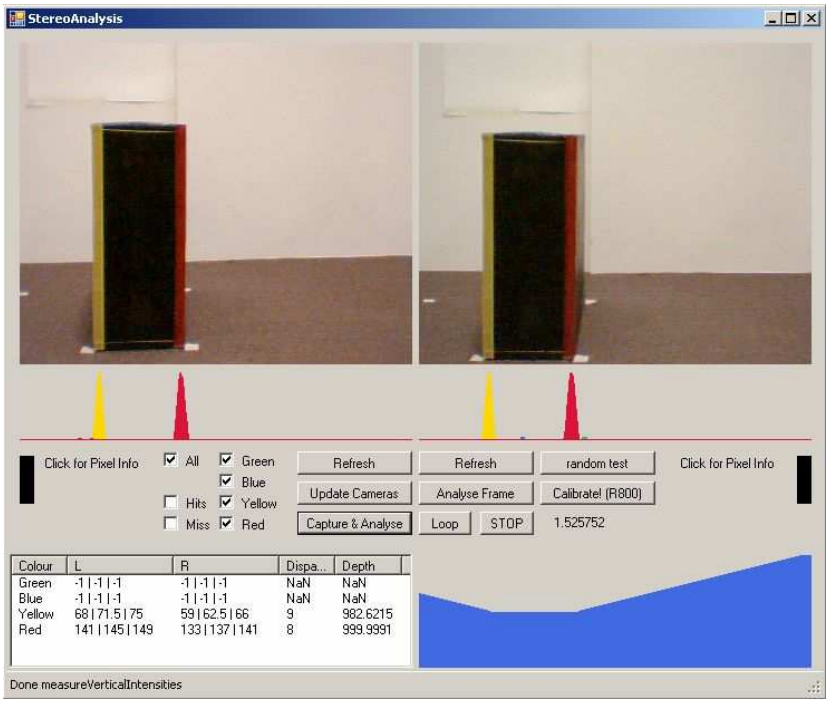


Figure 3.1: Stereoscopic Vision Component in action—box is 1m away from cameras.

under controlled conditions, and that the development of the system could be accomplished in the limited time available.

The single "obstacle" used was a specially marked-up box, which can be seen in Figure 3.1. The box was painted black, then electrical tape of varying colours was placed down each edge. The box was placed against a white backdrop to prevent confusion between an edge and other objects of similar colour in the scene. The experimental setup is discussed in more detail later in the chapter.

The software locates the edges of the box by applying a thresholding algorithm that identifies pixels with colour values that fall within a particular range of hues, saturations and brightnesses. The number of matching pixels found are then summarised according to their position on the x-axis. These results are rendered as a histogram, showing the likelihood that each column of pixels contains a particular edge. The y-axis positions are ignored, as it is only the horizontal disparity between the left and right images that provides the necessary information for calculating depth.

The software assumes that there is a maximum of one peak in each image for each colour, which is acceptable given the controlled experiment conditions. The horizontal position of each peak is taken as being the average between the

rise and fall of the peak. The values are considered to rise to a peak when they become greater than 50% of the maximum pixel count per x-axis position for that colour in each image.

3.1.4 Calculating depth from pixel disparity

Having determined the pixel disparity of each colour marker, the software must next calculate the estimated depth based on this. The pixel disparity of each marker is first converted into millimetres as measured on an imaginary image plane 500mm from the cameras. Simple trigonometry is then used to calculate the depth of each marker in millimetres.

The stereo calculations detailed in this section are based on an extremely simplified model of the world. It is important to consider the stereo system described in the context of the overall aim of this research, which is to test the combination of a stereo vision system and sonar. In order to keep the system simple and achievable in the time-frame, the stereo model assumes that the obstacle is positioned equidistant from the two cameras. In the real world, of course, this will usually not be the case, but the error that this introduces is minimal. This is discussed further in Chapter 4.

To determine the conversion ratio between pixels and millimetres as at 500mm away, an image was captured of a ruler placed 500mm in front of the camera. This showed that 100mm in the image equated to 108 pixels. Therefore millimetres can be calculated thus:

$$\begin{aligned} \text{mm} &= \text{pixels} \times \frac{100}{108} \\ &= \text{pixels} \times \frac{25}{27} \end{aligned}$$

Figure 3.2 shows the geometry involved. The two cameras are at A and G. They were set 135mm apart. They each have imaginary image planes 500mm away shown by FE and EI respectively. An object at H (the *known object*) is at a known depth of 800mm. Another object at D (the *unknown object*) is at an unknown depth of x. The difference in the disparity between the known object H and the unknown object D as observed in mm on the image plane is represented by δ. Thus:

Given δ and θ, find x.

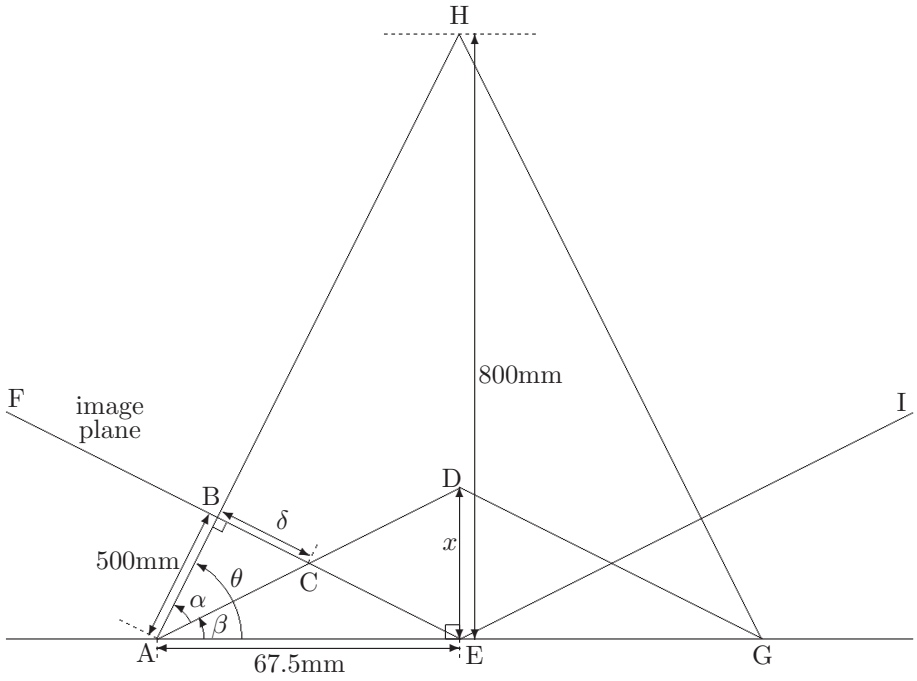


Figure 3.2: Depth calculation geometry

First consider △AEH:

$$\begin{aligned}\tan\theta &= \frac{EH}{AE}\\ \therefore \theta &= \arctan\left(\frac{800}{67.5}\right)\\ &= 85°10'\end{aligned}$$

Next consider △ABC:

$$\begin{aligned}\tan\alpha &= \frac{\delta}{AB}\\ \therefore \alpha &= \arctan\left(\frac{\delta}{AB}\right)\end{aligned}$$

Now x can be calculated using $\triangle$AED, thus:

$$\begin{aligned} x &= AE \times \tan(\beta) \\ &= AE \times \tan(\theta - \alpha) \\ &= AE \times \tan\left(\theta - \arctan\left(\frac{\delta}{AB}\right)\right) \\ &= 67.5 \times \tan\left(85°10' - \arctan\left(\frac{\text{pixels} \times \frac{25}{27}}{500}\right)\right) \end{aligned} \tag{3.1}$$

Thus, the depth (x) of an object in the scene can be calculated given the disparity between the perceived location of the object in pixels.

3.1.5 Calibration

As discussed in Chapter 2, there are two categories of calibration parameters that can be applied to vision systems—intrinsic and extrinsic. The particular cameras that have been used (Logitech Quickcam Express USB cameras) are designed for desktop use and whilst they have not been engineered with high precision in mind, they have fairly standard optical characteristics that might be found in most cameras. Due to the low level of accuracy required for this research, intrinsic parameters such as focal length and lens distortion are ignored.

Small changes in the orientation of the cameras relative to each other, however, can result in major changes in calculated depth. Extrinsic calibration of the relative orientation of the cameras is accomplished using the same geometry as described above. The formula for calculating the depth of an object point given its disparity (Equation 3.1) can be transposed to solve for θ, thus calculating the angle of the cameras relative to each other given a known depth and disparity measurement:

$$\begin{aligned} x &= AE \times \tan\left(\theta - \arctan\left(\frac{\delta}{AB}\right)\right) \\ \frac{x}{AE} &= \tan\left(\theta - \arctan\left(\frac{\delta}{AB}\right)\right) \\ \arctan\left(\frac{x}{AE}\right) &= \theta - \arctan\left(\frac{\delta}{AB}\right) \\ \therefore \theta &= \arctan\left(\frac{x}{AE}\right) + \arctan\left(\frac{\delta}{AB}\right) \end{aligned}$$

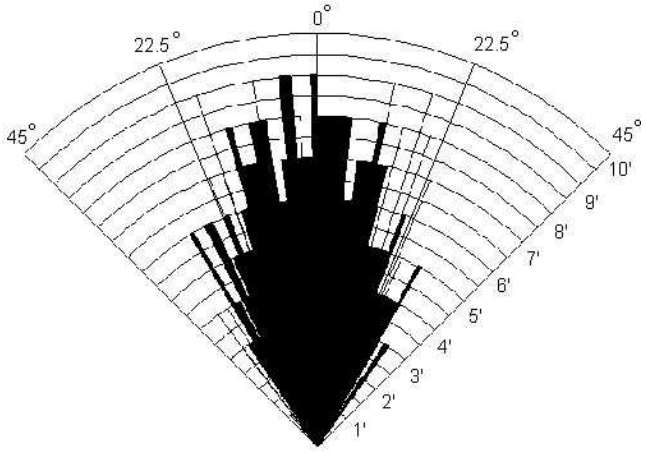


Figure 3.3: SRF04 Beam Pattern (Acroname Inc. 2004)

This calibration functionality was built in to the stereo vision component. It was used to calibrate the orientation of the cameras before each set of experiments by placing the box at a known distance, running an analysis and then using the known distance with the formula above to calculate the orientation of the cameras.

3.2 Sonar

Devantech's SRF04 Ultrasonic Ranger (Devantech Ltd 2003*a*) was chosen for sonar sensing, and is pictured in Figure 2.2 (Hall 2005). The SRF04 produces a pulse with a frequency of 40KHz. Devantech Ltd (2003*b*) specifies a working range of 3cm–3m with an accuracy of 3–4cm. The accuracy is affected by the inherent inaccuracies associated with sonar discussed in Chapter 2, as well as some particular characteristics of the design of the SRF04. For example, the echoes received by the SRF04 may come in the form of multiple wavefronts reflected by a selection of objects. The sensor reports a received echo when the first wavefront to reach a certain threshold is detected—this may be the second or third wavefront.

Acroname Inc. (2004) have conducted tests examining the beam pattern of the SRF04, as shown in Figure 3.3. This indicates a reliable working range of 7–8 feet or approximately 210–240cm. The beam width varies according to the distance, but averages approximately 45°.

Two SRF04 ultrasonic rangers were used in the setup. As shown in Fig-

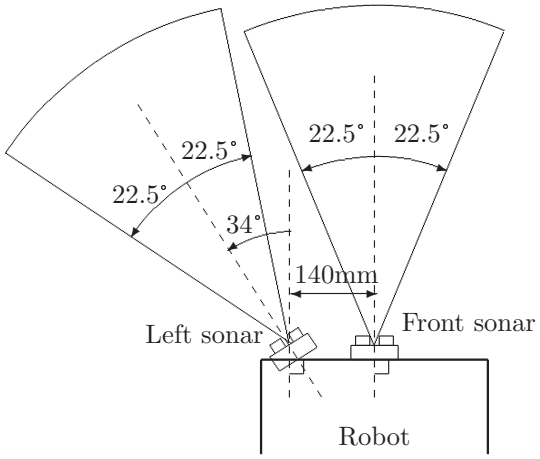


Figure 3.4: Sonar placement on the robot

ure 3.4, one sensor was placed at the front of the robot directly in the centre facing forwards, the second was placed 140mm to the left of the centre pointing to the left by 34°. Both were mounted on the upper level of the FRP (approximately 360mm above the ground) and were angled upwards by a few degrees to minimise the problem of detecting the floor as an obstacle (Devantech Ltd 2003*a*). This setup maximises the sensed area to the front and front-left of the robot whilst minimising the blind area between the sensors (consider that the blind area shown in Figure 3.4 is in fact smaller than it appears from the diagram due to the beam width being wider towards the sensor itself, as shown in Figure 3.3). The sonar sensor arrangement is optimised for detecting an obstacle to the left of the robot. Additional sonar sensors could easily be added, to the right for example, to make a more general system.

3.3 Compass

Obstacle avoidance is very easy indeed in the case where no other goal is present—the robot could simply remain stationary. In order that there be a reason for wanting to travel in the direction of an obstacle in the first place, an additional goal must be set. This study employed the use of a digital compass to set a goal in the form of a bearing for the robot to follow. To account for changes in the mounted orientation of the compass, an initial reading was taken at the beginning of each experiment for calibration. This calibration reading was then used as the target bearing for that experiment. The robot itself was carefully aligned with respect to the environment at the beginning of each experiment ensuring that the initial compass calibration reading accurately reflected the

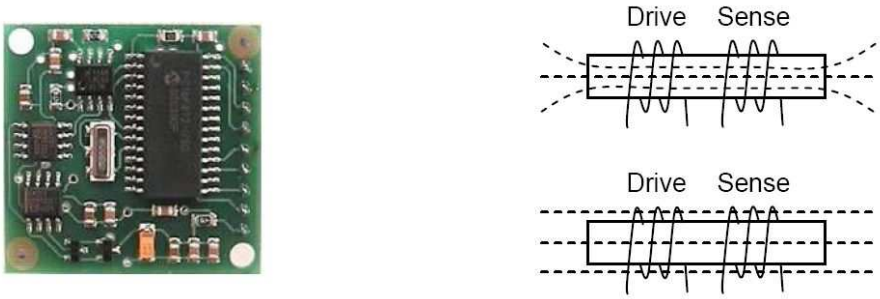


Figure 3.5: CMPS03 Digital Compass and flux in unsaturated (top) and saturated (bottom) core (Hall 2005)

desired direction, and was consistent between experiments.

The most commonly used type of digital compass for mobile robotics is a Fluxgate compass (Nehmzow 2003), which measures magnetic fields using a controlled electromagnet. Hall (2005) describes its operation thus:

> "A Fluxgate compass consists of a drive and sensing coil on a common core. By alternately driving the drive coil (altering the flux through the core) a voltage is induced in the sensing coil which varies depending on the ambient magnetic field. Two cores are required to sense north." (Hall 2005)

The CMPS03 digital compass (Devantech Ltd n.d.*b*) is a Fluxgate compass and is shown in Figure 3.5 along with a diagram of its operation. It uses two magnetic field sensors to compute the direction of the horizontal component of the prevailing magnetic flux (Devantech Ltd n.d.*a*). It is sensitive enough to detect the direction of the earth's magnetic field, however there are many other potential sources of magnetic fields which can interfere with readings. Devantech Ltd (n.d.*a*) reports an accuracy of 3–4° and readings are reported at a resolution of 0.1°. This means that differences of up to 0.1° can be detected, but any given reading may be up to 3–4° off the correct bearing.

Initial testing in the experimental environment revealed that there were major magnetic disturbances at the floor level, possibly due to electrical cabling under the floor. The robot itself is also a source of interference—consider the magnetic fields produced by components such as the wheel motors or the laptop computer. To minimise the interference from localised magnetic fields, the compass was mounted on a plastic pole some 96cm from ground-level.

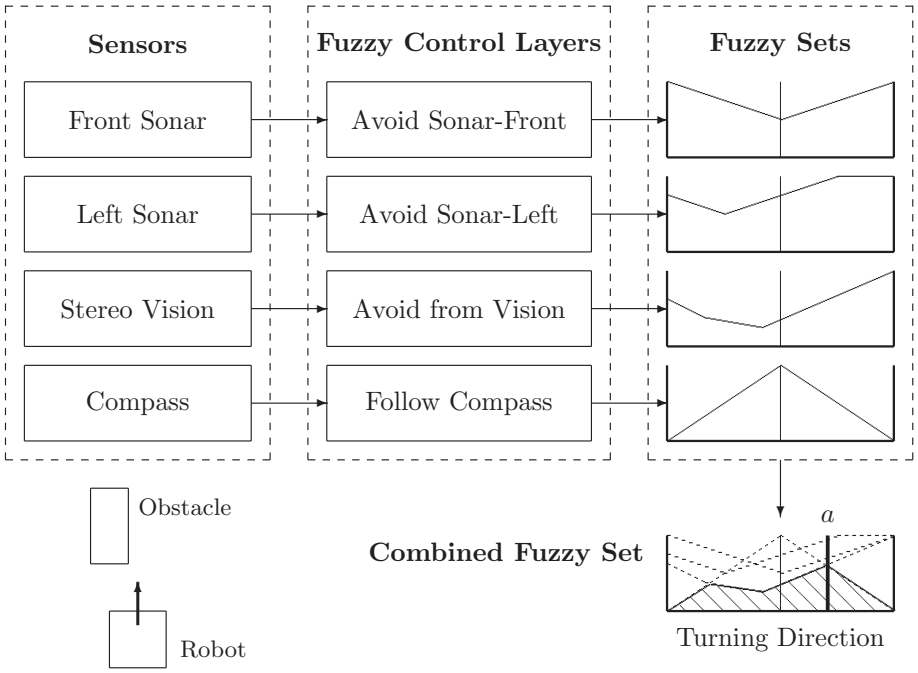


Figure 3.6: An example of the Dynamic Fuzzy Logic Control System Architecture: *a* indicates the crisp output from defuzzifying the combined fuzzy set—The robot turns to the right to avoid the obstacle.

3.4 Control System

The control system employed in this study was a new approach called the Dynamic Fuzzy Logic Control System Architecture, and is illustrated in Figure 3.6. The approach is based on elements from a Fuzzy Logic Control System as described by Yen & Pfluger (1995) and discussed in Chapter 2. It is a simplified Fuzzy Logic system that bypasses the fuzzy rules and fixed fuzzy sets of the classical approach. Instead, each behaviour is responsible for dynamically creating an appropriate fuzzy set representing the preferred outputs of that behaviour at each moment. In the system used in this study, there was only one fuzzy conclusion that was considered. This was the Turning Direction of the robot.

Sensor Fusion is accomplished as part of the command fusion process rather than being a preparatory step. Each sensor contributes to the final fuzzy set in one step. Whilst it would be possible for each fuzzy control layer to take input from as many sensors as desired, the system tested involved each layer taking input from one sensor.

Defuzzification was achieved by taking the first maximum value. The nature of the particular fuzzy layers being employed is such that the occurrence of

multiple maximum values in a combined fuzzy set is unlikely. When this does occur, however, it indicates that there is more than one outcome considered to be equally 'optimal' by the system. Only one optimal solution is required, and in essence, whichever one is chosen is not important. The system selects, then, the first maximum value that is found, searching left to right. This means that there is a bias present for turning to the left in these multi-optimal situations.

In order to use the system in experiments it was implemented in Java, as part of the same existing program that controls the robot. The Java classes pertaining to the Dynamic Fuzzy Logic Control System are shown in a class diagram in Figure 3.7. Firstly, an AbstractControlLayer class contains the functionality required to store the values in a fuzzy set. The fuzzy set itself is represented by 352 values in an array ranging between 0 and 200. 352 was chosen as the number of values in a fuzzy set because the width of the resolution of the cameras was 352 pixels, making the translation to a fuzzy set simple. 200 was chosen as a maximum value, nominally representing a range of 200cm. Each control layer extends the abstract class and is required to implement the updateValues() method. This method is the mechanism for taking readings from sensors and dynamically generating the values for the fuzzy set for each layer. FuzzyControlSystem is the main class of the system, encapsulating each of the control layers within it. This class also extends the AbstractControlLayer class, as it itself holds the values for the combined fuzzy set. The combined values are calculated by taking the minimum of the values across all the control layers at each position in the fuzzy set. This command fusion functionality is implemented in the updateValues() method of the FuzzyControlSystem class. The defuzzify() method finds the maximum value in the combined set as described above and outputs the position in the combined fuzzy set that this value is found at. The code for this system can be found in Appendix B.

Each layer type has a different method for producing its dynamic fuzzy set. The compass layer constructs its fuzzy set by rendering a peak at the desired compass direction, and sloping down on either side, wrapping around the edges.

The stereo vision layer takes its fuzzy set values from the stereo vision component. First it calculates the depth to markers in the scene as described earlier in the chapter. It then places values in the fuzzy set directly at the position and depth that they are detected at in the scene. The values between these points are filled in using a simple line drawing algorithm. The values at each far side are filled by drawing a line that stats from the marker at that position then slopes gently upwards until it reaches the maximum value of 200 or the far edge of the fuzzy set. The values are stored in a file so that they can then be ready by the Java program. For more detail please refer to the source code

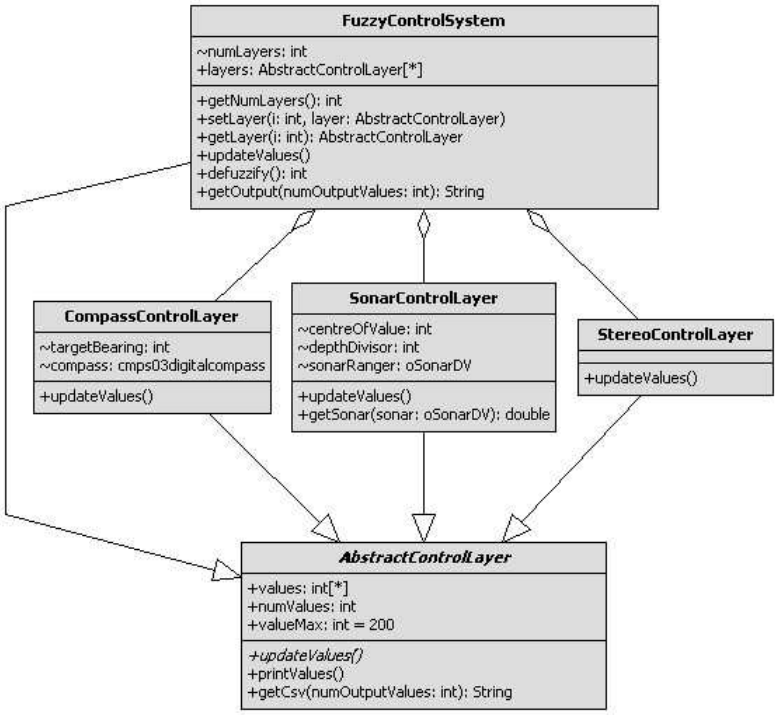


Figure 3.7: Class diagram of the Dynamic Fuzzy Logic Control System

of the stereo vision component in Appendix A.

Each sonar layer is created with a *centreOfValue* parameter that sets the position in the fuzzy set at which the sonar readings should be centred on. This is roughly equivalent to the angle at which the sonar is physically placed on the robot. To create the dynamic fuzzy set, the reading from the associated sonar sensor is recorded in the fuzzy set at the position specified by centreOfValue. A line sloping up on each side sets the remainder of the values. If the maximum is reached before the edge of the fuzzy set is reached, the maximum value (200) is used for remaining fuzzy set values. The front sonar's centreOfValue is set at 352/2; the centre of the fuzzy set. The left sonar's centreOfValue is set at 120 (approximately 352/3). The resulted in an appropriate amount of overlap between the two sonar sensors.

3.5 Experimental setup

This section describes the physical configuration of the robot and the experimental environment used to conduct experiments.

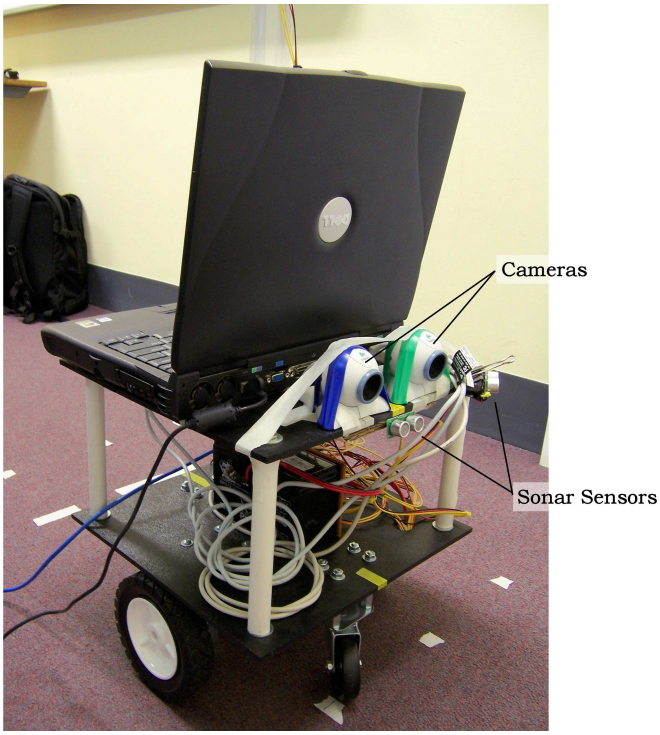


Figure 3.8: Configuration of the Flexible Robot Platform

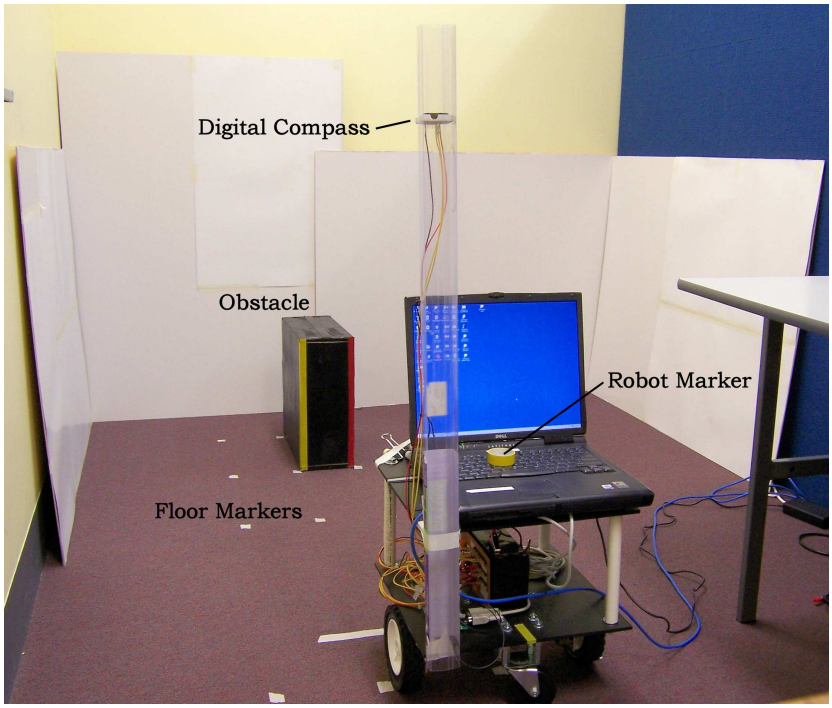


Figure 3.9: The experimental environment all tests were conducted in

Figure 3.10: Aerial camera mounted in a light fitting for measuring the position of the robot during experiments

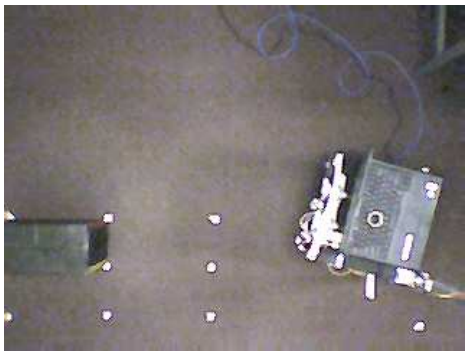


Figure 3.11: Sample image taken by the aerial camera

The Flexible Robot Platform (FRP), developed by Hall (2005), was used to conduct the experiments in this study. It is pictured in Figure 3.8 as configured for the experiments. The robot was designed to be easy to reconfigure for different purposes, and this study is one example of that. The camera and sonar positions are indicated.

The particular cameras that have been used are a pair of Logitech Quickcam Express USB cameras. Being domestic multi-purpose desktop cameras, they are not designed to distort the lens in any unusual way. By the same token, however, they are not engineered with high precision in mind either.

The object used as the obstacle for the robot to avoid was a cardboard box measuring 38cm x 16cm x 38cm. It was spray-painted black to improve the accuracy of the vision system. Coloured tape was attached to each vertical edge for the vision system to detect. The box was sturdy enough to stand solidly in position and last the year that this study ran over, but light enough for the robot to push it along the floor in the event of a collision.

The experiments were conducted within an experimental environment that allowed the positions of the robot and the obstacle to be placed at standardised locations. The setup is shown in Figure 3.9. The FRP is pictured as well as the obstacle used for experiments. White boards were placed as a backdrop to prevent the vision system from becoming confused by the yellow or blue walls that can be seen in the background. Markers on the floor allow accurate positioning of objects in the scene. This photo also shows the digital compass, which is mounted on a plastic pole for reasons discussed in section 3.3.

An aerial camera was mounted in a light fitting above the setup, as shown in Figure 3.10. This was used to measure the position of the robot at each frame of the experiments. The position was measured by identifying the robot marker in the image. This is a yellow roll of tape positioned in the very centre of the

robot. It can be seen in Figure 3.9, and also in Figure 3.11, which is an example image taken by the aerial camera during an experiment.

3.6 Sources of error

As with all experimental work, there are various sources of error that exist in each component of the system. These must be considered before any experiments can be conducted. One motivation behind gaining an understanding of the error present in experiments is that it can reveal ways in which the error can be minimised. However, many sources of error are unavoidable and cannot be eradicated. Nevertheless, understanding the error enables it to be taken into consideration when examining the results of experiments. There are two categories of error typically used to model the error present in experiments. These are random error and systematic error.

"Random error is any type of error that is inconsistent or does not repeat in the same magnitude or direction except by chance." (Viswanathan 2005) Random error is present when a device used to take measurements gives readings that vary over multiple iterations of the same experiment. Random error can be dealt with by repeating experiments and statistically analysing the results.

"Systematic error is any error that has a consistent effect." (Viswanathan 2005) For example, values given by a measurement device that are always incorrect by a fixed amount exhibit a systematic error. Systematic error is more difficult to account for than random error. It relies on the experimenter thinking of each cause and either correcting for it or discussing it.

The experiments conducted as part of this study are subject to both random error and systematic error. The rest of this section discusses specific errors that require consideration.

3.6.1 Vision

There are computer vision related sources of error present in the stereoscopic vision subsystem as well as the images captured by the aerial camera to document experiments. Quantization due to the rasterized nature of digital images, camera optical properties, and the exact positioning of the stereo cameras are all sources of unavoidable errors.

There is random error present due to the rasterized nature of digital images and the quantized view of the world that results. The accuracy of measurements taken via digital imaging is therefore fundamentally dependent on the resolution of the input images. As discussed in section 3.1.4, the resolution of the Logitech Quickcam Express cameras is such that 108 pixels in the image corresponds

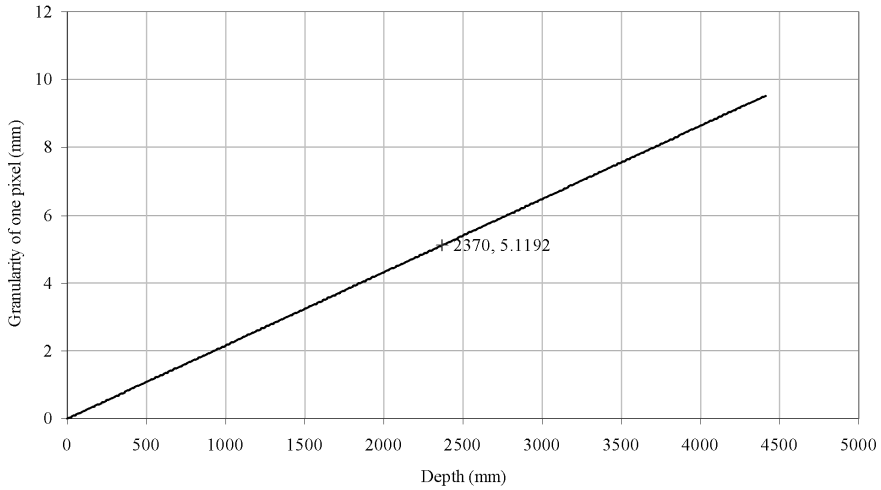


Figure 3.12: Width of a single pixel in the real world at various depths. The distance between the aerial camera and the top of the robot is indicated.

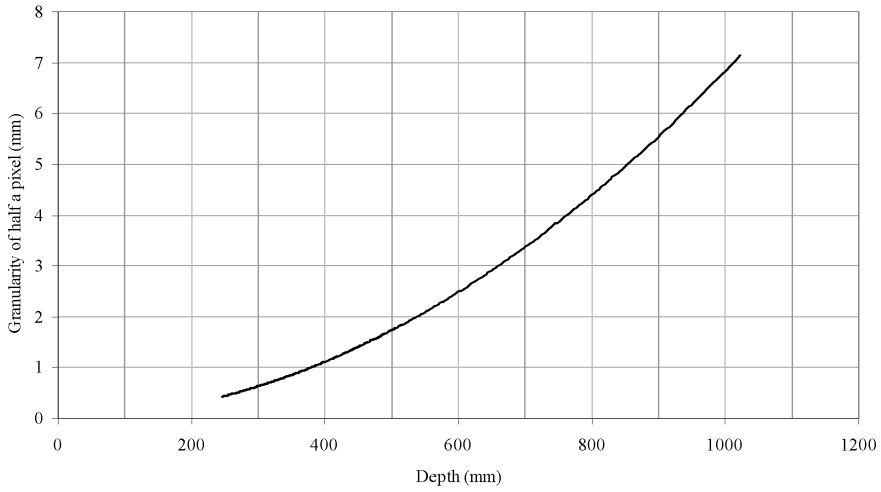


Figure 3.13: Depth represented by half a pixel of disparity at various depths

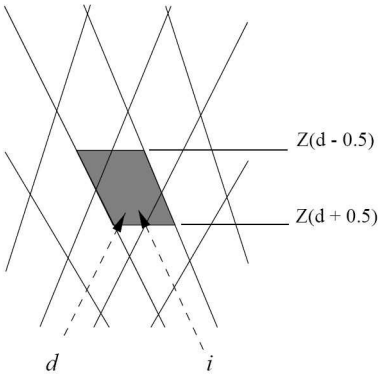


Figure 3.14: Region of uncertainty for a given pixel i and disparity d (Murray & Little 2000)

to 100mm in the world when measured 500mm from the cameras. This can be extended using trigonometry to determine the real-world width of a single pixel at various distances from the camera. This relationship is illustrated in Figure 3.12.

The height of the aerial camera above ground level is 2730mm. The marker used to measure the position of the robot from images taken by this camera is on the robot at a height of 360mm. The distance from the aerial camera to the marker is therefore 2370mm. As indicated in Figure 3.12, objects at this distance from the camera have a resolution of approximately 5.1mm per pixel. This results in a random error of up to 5.1mm in the measured position of the robot in each frame.

Similarly, the depth of an object in a stereo pair determines the depth represented by each pixel of disparity. There is therefore a region of uncertainty as to the position of an object in real space, as illustrated in Figure 3.14 (Murray & Little 2000). Given a pixel of interest as viewed from one camera i and a disparity measured from a second viewpoint d, an inaccuracy in the calculated depth could be as much as the real-world equivalent of ± 0.5 pixels. Figure 3.13 shows that the depth represented by each pixel (or half-pixel) of disparity grows exponentially as the total depth increases. Half-pixels were chosen for this graph as the stereoscopic vision component averages the positions of the two edges of identified markers, resulting in a disparity resolution of 0.5 pixels (described in 3.1.3).

3.6.2 Sonar

There are inherent problems associated with sonar including imprecision, specular reflections and crosstalk, as discussed in section 2.3.1. The SRF04 sensor has a specified working range of 3cm–3m and an accuracy of 3–4cm as described earlier in this chapter. Sonar readings are also subject to random error from noise in the electronics.

3.6.3 Compass

As discussed previously in section 3.3, the cmps03 digital compass has a resolution of 0.1° and an accuracy of 3–4°. As with the sonar sensor, the compass is subject to random error from noise in the electronics.

The environment in which experiments were conducted was subject to local magnetic fields that interfered with the detection of the earth's magnetic field by the compass (see section 3.3). This problem introduces a systematic error into the measurements taken. There are many potential sources of magnetic interference, most notably other electrical devices and wiring, and the robot itself. It was impractical to move the experimental setup to a location with no electromagnetic interference—such a place would be difficult (if not impossible) to find, in any case. To minimise the effects of this, the compass was mounted 96cm above the ground.

3.6.4 Wheels

The robot itself is subject to random and systematic error when moving due to wheel slippage and general inaccuracy in the degree to which the motors can be finely controlled. Differences in floor surfaces can greatly affect the degree to which a robot's wheels slip (Murphy 2000).

This can affect the robot in two ways. It can affect the robot's ability to travel in a straight direction. It can also affect its ability to travel the same distance when told to travel at a certain speed for a set amount of time.

3.6.5 Error handling

The nature of systematic error is that there are always likely to be factors that are overlooked. Care has been taken, however, to ensure that the major sources of systematic error were considered, and minimised and/or accounted for where possible.

Random error is accounted for by repeating key experiments multiple times and averaging the results. Standard deviation of a sample can be used to consider the spread of individual values. Sample standard deviation is used to

estimate the standard deviation of a population, as experiments represent a sample of an infinitely large set of possible experiment repetitions. Standard error is used to estimate the standard deviation of the sample mean. Standard error is calculated thus:

$$\text{standard error} = \frac{\text{standard deviation}}{\sqrt{\text{number of measurements}}}$$

In general, the overall effect of sensor error on the control decisions made by a fuzzy control system should be minimal (Yen & Pfluger 1995). Such is the advantage of a system designed to handle vagueness and uncertainty (see section 2.4.3).

3.7 Experiment Plan

The experiments conducted are ultimately designed to test the hypothesis—that better obstacle avoidance can be achieved by combining stereoscopic vision and sonar using a fuzzy logic control system, as opposed to using these sensor types individually. In order to test this it is first necessary to conduct experiments examining the accuracy of the various components that make up the whole system.

In this section, the sources of error associated with each component are studied. First, the stereoscopic vision subsystem is examined. This is followed by tests that consider the error associated with the compass. Finally, experiments are conducted that test the robot's ability to avoid an obstacle whilst following a compass bearing using sonar, vision, and the combined approach.

3.7.1 Stereo vision subsystem

The vision subsystem is an important part of this study. Whilst it is secondary to the primary focus of the control system, it is important to examine its accuracy and reliability in order to ensure the validity of the main results. Two experiments were conducted that focussed on the vision subsystem. First the accuracy of depth estimations was assessed from a single calibration. Then the affect of lateral displacement on depth estimation was examined. The obstacle's red marker was used for all vision experiments

In the first vision experiment, the marker was placed at different positions along a line extending from the centre of the front of the robot. The system was first calibrated by placing the marker at a distance of 600mm and following the calibration procedure described in section 3.1.5. The marker was placed at distances of 200mm–1400mm at 100mm intervals. The depth was measured by

the vision system six times for each distance. The average depth was calculated for each distance, along with the standard error.

In the second vision experiment, the marker's position was varied laterally. Tests were conducted at distances of 600mm–1000mm at 100mm intervals. At each distance the system was calibrated with the marker 0mm off-centre. The marker was then moved directly sideways in 50mm increments in both directions until no longer viewable by both cameras. In order to standardise results, Pythagoras' theorem was used to calculate the displacement to each position based on the measured (diagonal) distance. The average displacement was calculated for each depth along with the standard deviation and standard error.

3.7.2 Compass error

The compass was used in later experiments to provide a standard goal alongside obstacle avoidance. It was important that the error associated with compass readings be understood. As discussed in section 3.6, both random error and systematic error are potential sources of error when taking compass readings.

Tests were conducted to evaluate both types of error. It was important to establish how the error varied according to the physical position of the robot within the experimental environment. Tests were conducted in an area covering the expected path of the robot in the obstacle avoidance experiments. The robot was placed at 30cm intervals in a straight line extending from the starting position. At 90cm of depth it was then placed at 30cm intervals along a line to the right. At each position, 100 readings were taken in order to estimate the random error. The whole experiment was repeated in order to account for variations in the precise positioning of the robot at each stage.

3.7.3 Obstacle avoidance

The obstacle avoidance tests were conducted next. The ability of the Dynamic Fuzzy Logic Control System to combine stereoscopic vision and sonar was tested by running three sets of experiments. All tests involved the robot moving over 26 frames. The robot was programmed to calculate the direction and magnitude of its next turn-factor, then move for 500ms at a speed of 50 (out of 127—roughly 12cm/sec). An image was captured using the aerial camera while the robot was stopped after each movement. The sequence of images was then used to plot the path of the robot for each experiment. For each frame, an evenly spaced sample of ten values (from 352) from each layer of the control system was recorded. This allowed the decision made by the robot in each frame to be analysed. First baseline tests were conducted, followed by tests with the obstacle placed 100cm away then 60cm away.

The baseline tests were run in order to establish a point of reference to compare subsequent tests against. They did not involve the use of the obstacle. The robot was first run with only the compass fuzzy control layer active. This was done in order to establish the effects of local magnetic fields on the path of the robot when run in the test environment. The second baseline test employed the use of the compass layer and the sonar layer. This was done in order to establish what effects, if any, the testing environment had on sonar readings. This ensured that it was later possible to distinguish the results of the sonar detecting the actual obstacle from the results of the sonar reacting to the presence of other elements such as the walls or floor.

The obstacle was then placed 100cm away from the front of the initial position of the robot. The right edge of the obstacle (as viewed by the robot) was aligned with the centre of the robot. Three tests were conducted. The system was first run with the compass and two sonar layers only, then with the compass and vision layers only. Finally, the compass, sonar, and vision layers were all used in combination. These three tests were each conducted four times in order to measure the random error present. The choice of four repetitions was a compromise between the time available and conducting as many tests as possible to get the best estimation of error.

Finally, the obstacle was placed 60cm away from the robot. The three tests were conducted as per the 100cm tests, except that each test was conducted once only. The 60cm tests serve to show that the system is able to avoid an obstacle at an initial distance other than 100cm.

Chapter 4

Results and Discussion

This chapter presents the results of the experiments described in Chapter 3. This includes testing of the stereo vision subsystem and the compass in order to estimate the error present in these sensor types. The results of testing obstacle avoidance with stereo vision, sonar, and the combination of both are presented. The meaning of these results is then discussed in the context of the hypothesis.

4.1 Stereo vision subsystem

The stereoscopic vision subsystem was tested to determine its accuracy when calculating the depth of an object. Tests were conducted that examine the effect of changing the position of the test marker laterally and forward/backwards. The effect of the error measured on the obstacle avoidance system is then discussed.

4.1.1 Depth error

Figure 4.1 shows the average measured distance to the marker at a range of depths. The marker was placed at 600mm for initial calibration. As can be seen from this graph, there is a systematic error present in the results. The system underestimates the depth as the actual distance increases from the calibrated depth, and it overestimates the depth for distances closer than the calibrated depth.

The most likely cause of this error lies in the distance between the two cameras. Consider again Figure 3.2, which shows the geometry involved in the stereo calculations. It can be seen that even a small inaccuracy in the distance between the cameras will result in significant errors in the calculated depth. The calibration procedure (as described in section 3.1.5) adjusts for the relative

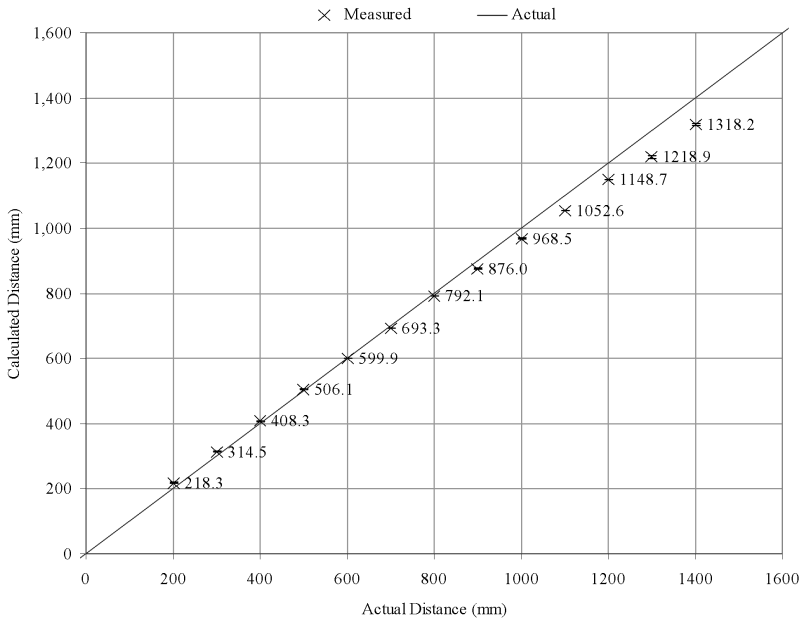


Figure 4.1: Average measured distance to target at a range of depths. The system was calibrated using a depth of 600mm. Error bars show standard error.

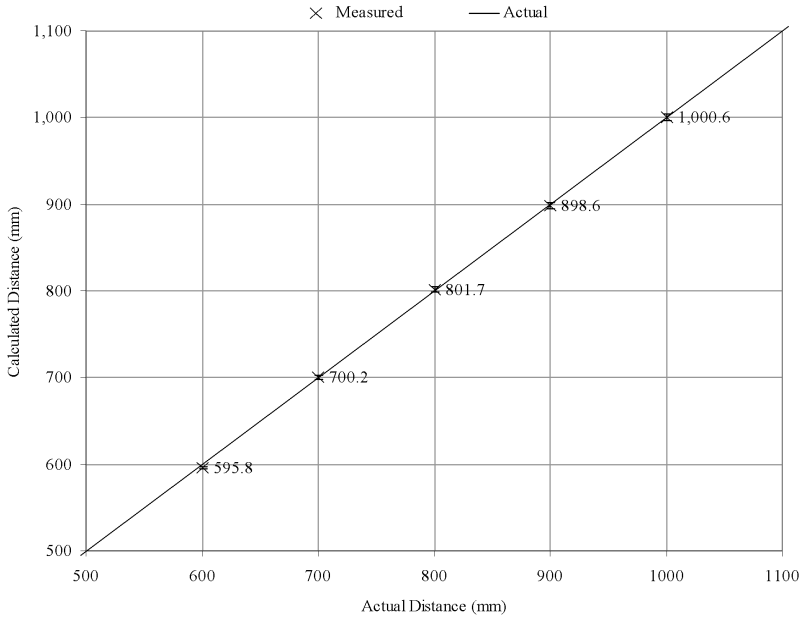


Figure 4.2: Average measured distance to laterally-spread targets. The system was calibrated for each distance with the target placed at the centre point. Error bars show standard error.

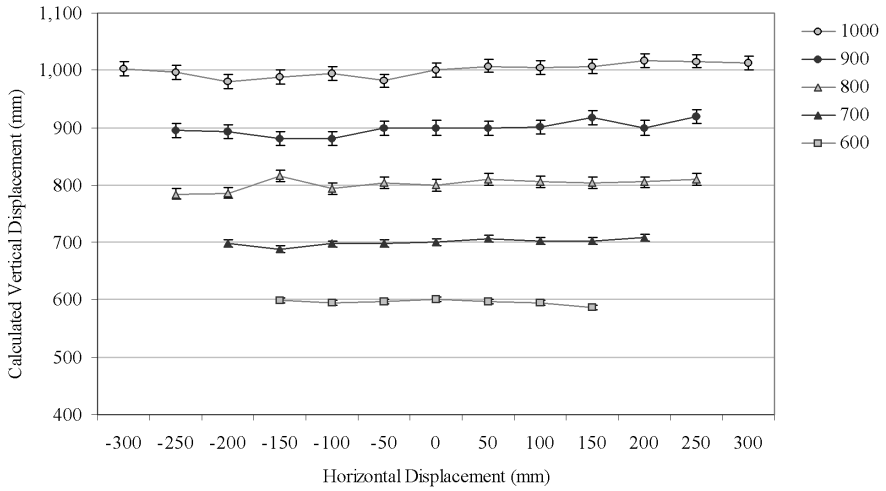


Figure 4.3: Lateral vision system error. Error bars show standard deviation calculated on all values for that depth.

angle of the cameras only. It assumes a known distance between the cameras. Thus, if the cameras move slightly with respect to each other, the calibration procedure as it stands will not correct for this.

The results from the lateral tests are shown in Figure 4.3. Each point in this graph represents an individual reading of the position of the marker. The values should ideally sit on the line for their respective depth. The average measured depth of the readings for each series is shown in Figure 4.2. This graph shows that the on average, readings deviate from the actual depth by only a few millimetres. Random error is estimated using the standard error across the readings for each depth. Localised fluctuations that can be seen in Figure 4.3 are also an indicator of the random error influencing readings. The error bars in this graph show standard deviation from the mean.

4.1.2 Ramifications

The main source of error present in the stereo vision system is the systematic error in depth measurements. The error increases as the distance from the calibrated depth increases. This occurs in both directions, with the system overestimating the depth of close objects and underestimating the depth of far objects. However, as this is likely dependent on the distance between the two cameras in the stereo pair, this relationship will vary according to the precise positioning of the cameras. The calibration procedure was carried out at the

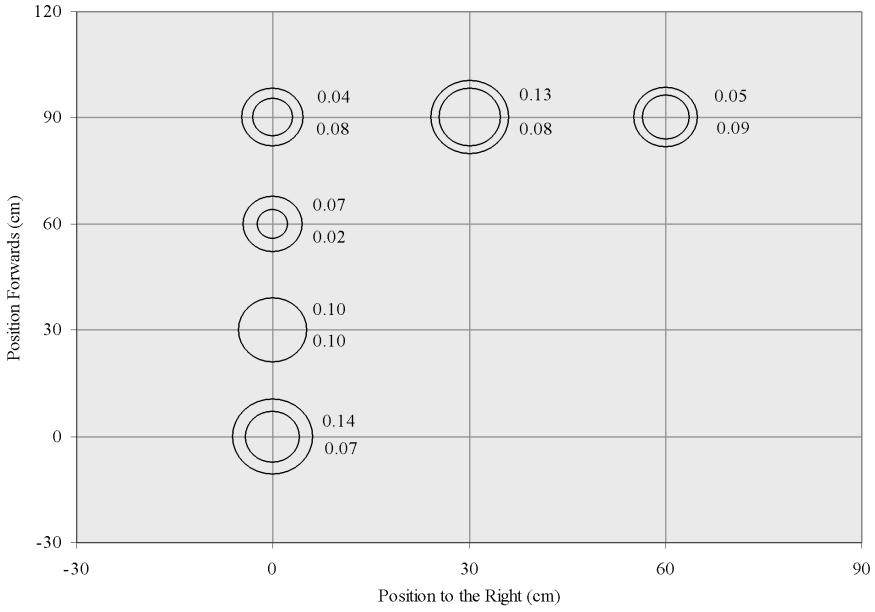


Figure 4.4: Standard deviation in compass readings at each test location

beginning of each obstacle avoidance experiment to minimise the error.

In the obstacle avoidance tests discussed later in this chapter, the calculated depth of the obstacle is used to specify the activation level that determines the values in the fuzzy set. This could be roughly equated to urgency. If the obstacle is near, the urgency of avoiding it is greater than if the obstacle is distant. This vague concept of urgency or nearness is represented well by a fuzzy logic control system. As discussed in section 2.4.3, errors present in the sensor data that is supplied to a fuzzy logic control system will generally have only a minor impact on the final control decision made.

4.2 Compass error

The compass experiments consider compass error. In particular, the question of how compass error varies according to spatial diversity is addressed. The measurements taken are used to estimate the random error associated with compass readings as well as the systematic environmental effects.

The random error can be estimated using the standard deviation of multiple readings taken at a particular point. How this varies across the test environment is illustrated in Figure 4.4. Each circle's area represents the standard deviation of a test consisting of 100 readings. The numbers indicate that there is generally

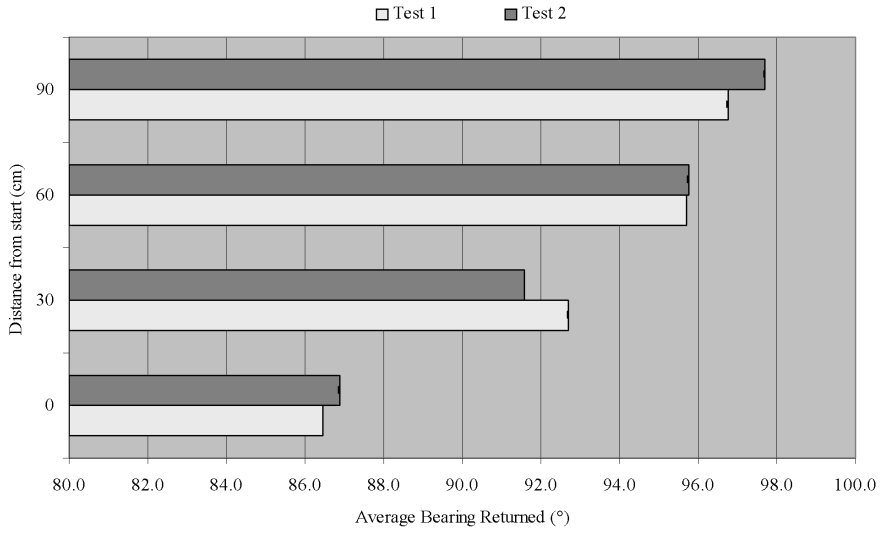


Figure 4.5: Average bearing returned by compass—forward/backward diversity

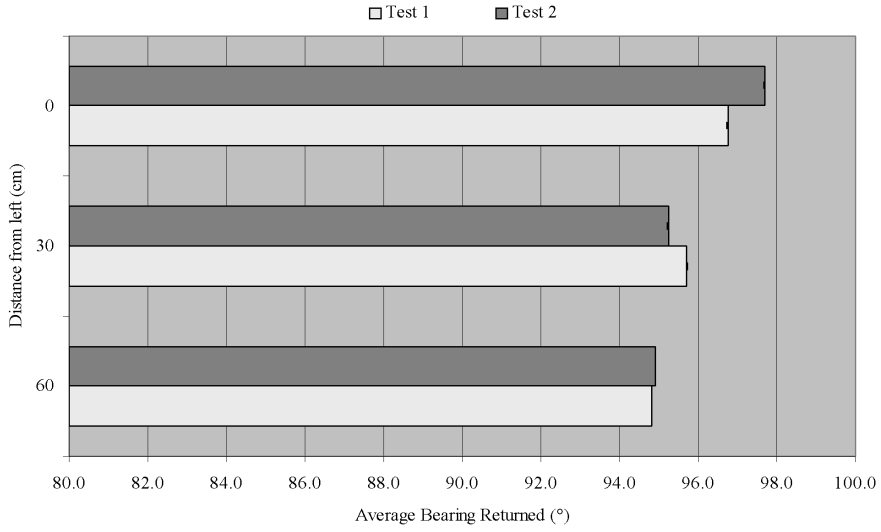


Figure 4.6: Average bearing returned by compass—lateral diversity

little variation (less than 1°) within each set of readings. It is interesting to note that some locations appear to produce more random variation than others. This could possibly be due to interference from electrical cabling or devices, which may not produce a consistent electromagnetic field.

Having considered the degree to which measurements fluctuate within a fixed location, the variation in the actual bearing returned between locations must be examined. Figure 4.5 shows the variance in the returned compass bearing as the robot is moved forwards. Error bars show the standard error, which is so small as to be comparatively negligible. As shown, there is a variance of approximately 10° between the starting position and 90cm forwards. In the case of the robot attempting to follow a compass bearing, this would result in the robot travelling in a direction 10° to the left with respect to the starting direction.

Figure 4.6 shows the variance in the returned compass bearing as the robot is moved to the right from the 90cm position of the previous chart. The two "0cm distance from left" tests in Figure 4.6 are the same as the "90cm from start" tests in Figure 4.5. This test is common to both directions of diversity and is therefore shown on both graphs. Figure 4.6, then, shows that the lateral diversity at this distance does not vary as much as the forward/backward diversity as described in the previous paragraph. The variance is in fact only 3°.

The compass readings fluctuate by as much as 10° depending on the position of the robot within the environment. The expected effect of this would be that the compass control layer would be instructing the robot to turn up to 10° too far to the left by the end of the experiment. This was confirmed in the baseline tests which are discussed in the next section. The obstacle was therefore placed to the left of the robot, forcing it to make the decision to avoid the obstacle by turning to the right. Thus eliminating the possibility that the compass readings might assist the robot in its obstacle-avoiding behaviour. If anything, the control system is now required to avoid the obstacle *in spite of* the compass' tendency to steer the robot towards it.

4.3 Obstacle avoidance

This section presents the results of the experiments testing the control system as applied to obstacle avoidance. First the results of baseline tests are discussed. Then the results of using the control system to avoid obstacles using stereo vision, sonar, and a combination of both are presented. A discussion of the meaning behind the results follows.

All of the aerial images captured for all experiments can be found in Ap-

pendix D. The control commands and the fuzzy sets that led to them can be found in Appendix E. Appendix C contains tables of measurements.

4.3.1 Baseline tests

Figure 4.8 shows the position of the robot recorded for each frame of the two baseline tests. Each point represents the centre of the robot at each frame. The size of the robot is indicated by the dotted outline at the starting position. As expected on the basis of the results discussed in the previous section, the robot drifts to the left when attempting to follow a fixed compass bearing. The positions that the obstacle was placed in for the next two experiments is indicated. Note the way the compass-following path collides with both positions.

The second baseline test used the sonar layers in the control system as well as the compass layer. This path is not dissimilar to the path that resulted from the compass-only behaviour. This shows that the sonar was not reacting in any significant way to other objects in the environment such as the walls and floor. Reactions produced when the obstacle is added to the setup are therefore solely due to the presence of the obstacle.

4.3.2 Main tests

The system's ability to avoid obstacles using stereo vision, sonar, and the combined approach was first tested with the obstacle placed 100cm from the starting position. Figure 4.9 shows the average positions of the robot over four repetitions of each test. Error bars indicate standard error. The initial position of the robot is indicated by R_0.

First consider the path of the robot when only the vision layer was used (together with the compass, as per all experiments). The robot initially turns to avoid the obstacle but then ultimately collides with it in frame 18. The outline of the robot in this position is indicated at R_1. To investigate why the collision occurs, it is instructive to examine the fuzzy sets output by the stereo vision layer for this experiment. The first eight frames of one of the test runs are shown in Figure 4.7. This shows that the obstacle was detected in the first four frames, but could no longer be seen from frame five onwards. Once the obstacle was out of the field-of-view of the cameras, the control system took instructions only from the compass. The result was a collision.

Next consider the path resulting from the sonar test runs. Figure 4.9 shows that the robot was able to navigate around the obstacle successfully. The outline of the robot indicated by R_2 shows the distance the obstacle was cleared by. Note, however, that the robot did not really start to turn until frames 8–10. Figure 4.11 shows the interaction between the fuzzy layers along with the final

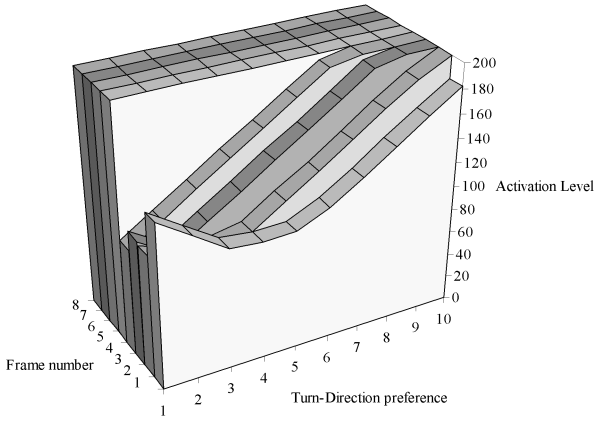


Figure 4.7: Stereo vision layer fuzzy set outputs from the first eight frames of the 100cm-obstacle vision-layer-only test

combined output fuzzy set for selected frames of this experiment. Figure 4.11d shows that the robot began to turn slightly in frame 6, however significant turning is not established until frame 8. Once the robot begins to turn, it continues on to avoid the obstacle.

The test of most interest is the combination of stereo vision and sonar. As with the stereo vision test, the combined approach resulted in early turning away from the obstacle, as can be seen in Figure 4.9. The average path of the combined approach diverges from the stereo vision path at approximately frame eight, however. Rather than continuing on to a collision, it continued to turn away from the obstacle.

Figure 4.12 explains the decisions the control system made during the first eight frames of the experiment. In the first frame, shown in Figure 4.12a, the vision system detected the obstacle and heavily influenced the control system's decision on which way to turn. The source of the reading in the left sensor was likely the wall to the left. It had no impact on the control decision, and decreased in the second frame after one turn to the right, in any case. In frames 2–4, the vision system's fuzzy set remained responsible for effecting turns to the right. The sonar sensors also began to detect the obstacle in these frames, but were not yet contributing to the control decision. In frame 5, the vision system could no longer detect the obstacle. This is the point at which the experiment that uses only vision no longer had any way of detecting the obstacle, which ultimately resulted in a collision. As this experiment also used the sonar sensors, however, the system was able to continue to react to the obstacle's presence. Frames 5–8 show how the two sonar layers resulted in the robot continuing to

avoid the obstacle. Note, too, the way the compass layer shifted its values to the left as the robot turned away from the desired heading.

In order to confirm that the system was not reliant on the obstacle being in a particular position, the three tests were also conducted with the obstacle placed 60cm away from the starting position. The path travelled by the robot in each of the three tests is shown in Figure 4.10. The starting position is again shown for scale (R_0). As per the 100cm-obstacle experiments, the stereo vision layer, when used alone, caused the robot to collide with the obstacle (at R_1). Due to the fact that the obstacle was closer to the robot to begin with, the sonar sensors detected it from the very first frame, as shown in Figure 4.13.

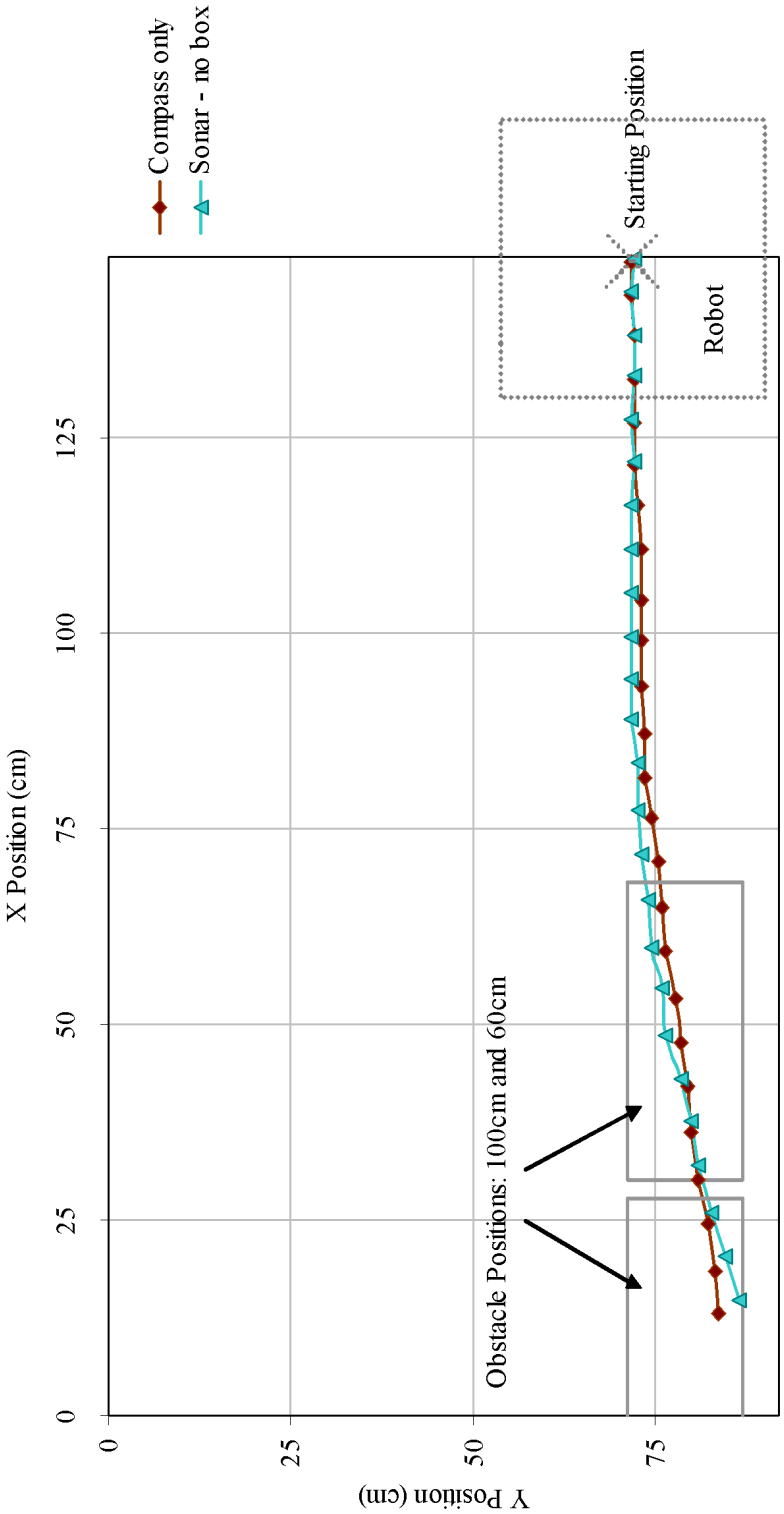


Figure 4.8: Compass-only and sonar - no-box tests for baseline

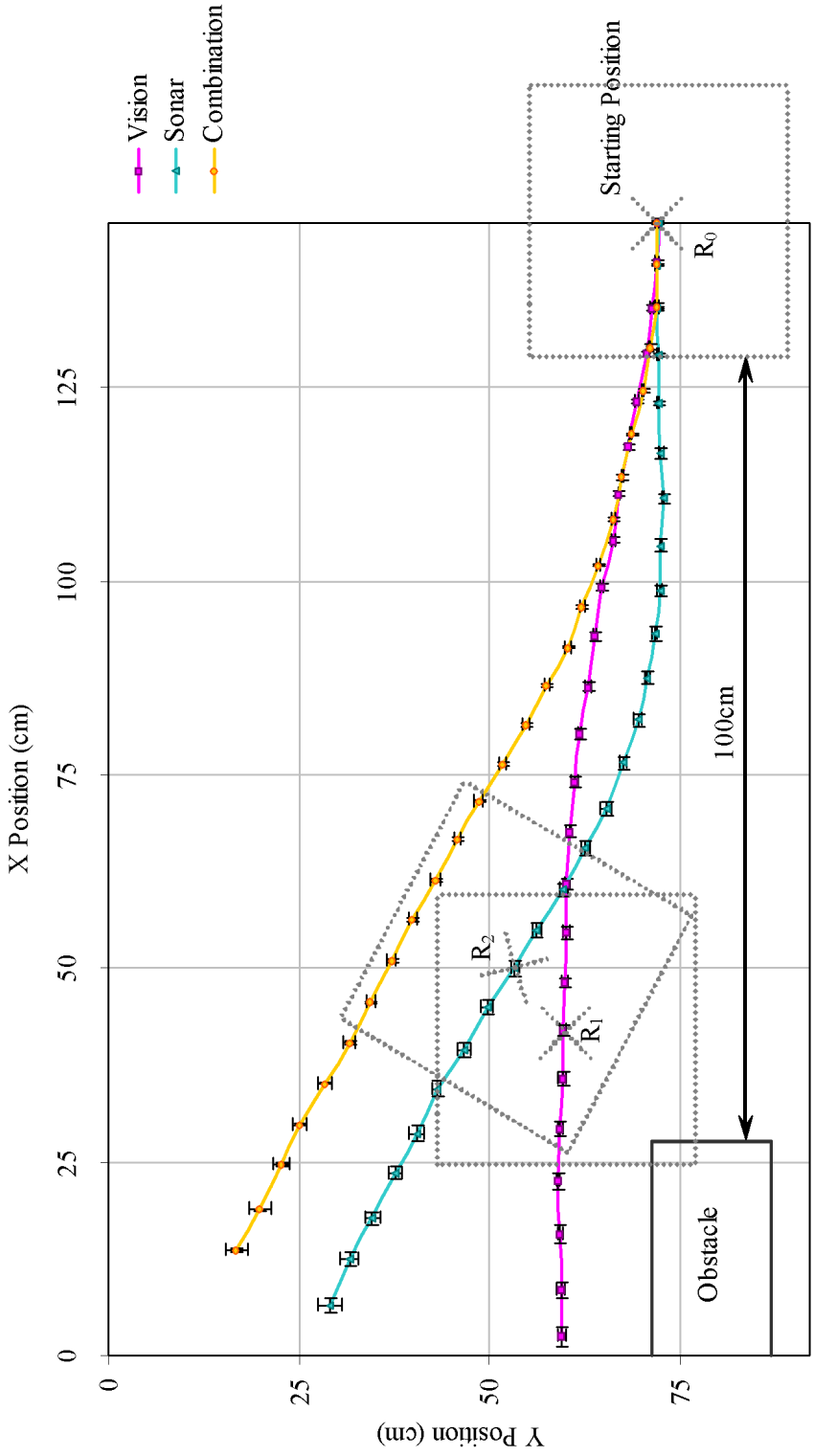


Figure 4.9: Average robot movement using vision, sonar, and the combined approach with the obstacle placed 100cm away. Error bars indicate standard error.

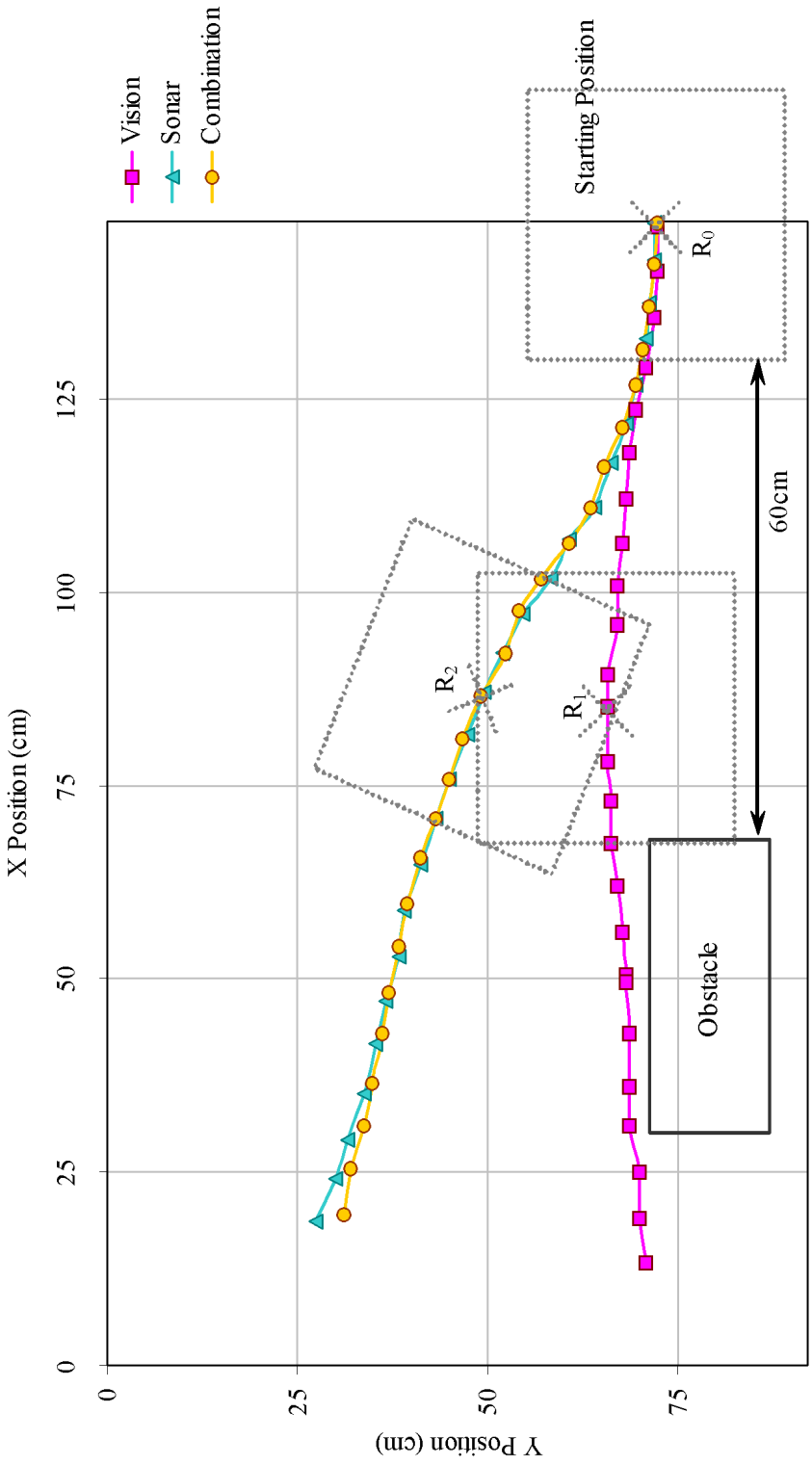
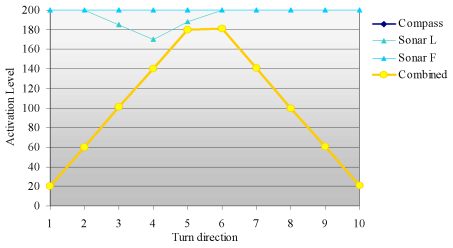
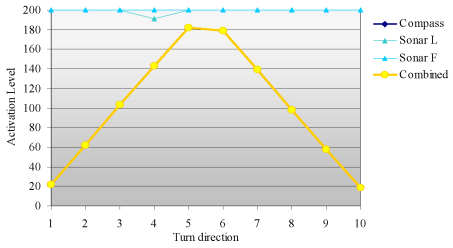


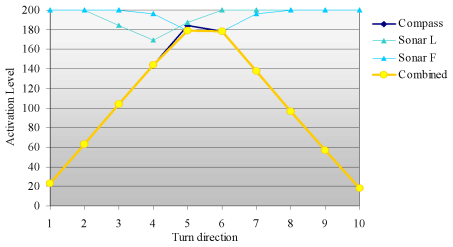
Figure 4.10: Robot movement using vision, sonar, and the combined approach with the obstacle placed 60cm away



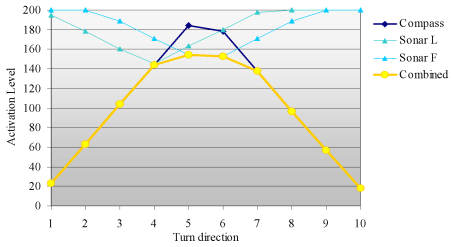
(a) Frame 1: Sonar cannot sense obstacle (interference from the wall is detected)



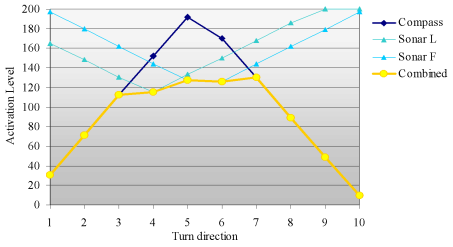
(b) Frame 2: Obstacle still out of sonar range



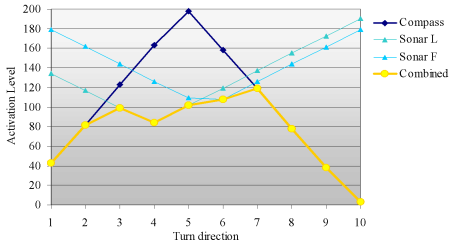
(c) Frame 4: Obstacle detected by both sensors; turn direction not yet affected



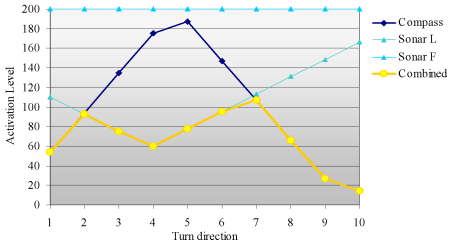
(d) Frame 6: Starting to turn to avoid*



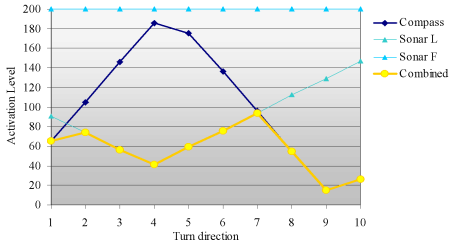
(e) Frame 8: Turning more sharply



(f) Frame 10: Still turning

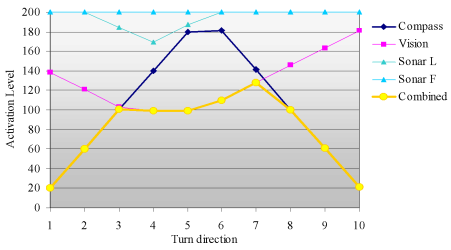


(g) Frame 12: Obstacle no longer detected by front sonar

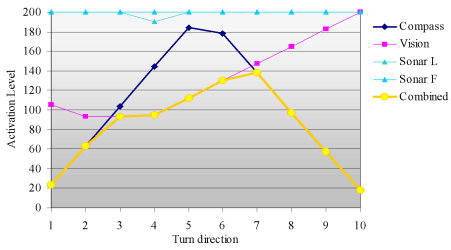


(h) Frame 14: Left sonar continues to avoid obstacle

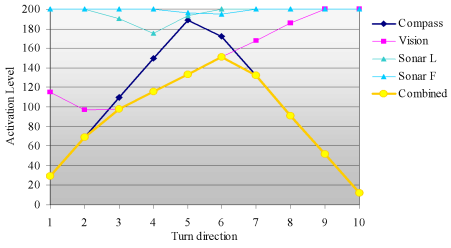
Figure 4.11: Interaction of Dynamic Fuzzy Sets during selected frames of an obstacle avoidance experiment using the two sonar fuzzy control layers. In this experiment the obstacle was placed 100cm away.
*These graphs only show ten samples out of 352 actual data points. The detail of the combined fuzzy set therefore does not sometimes show every peak.



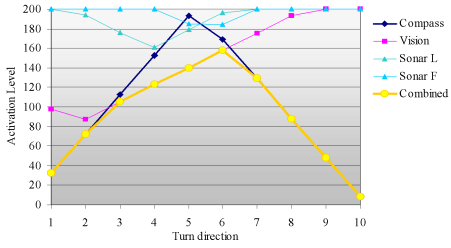
(a) Frame 1: Vision is in control



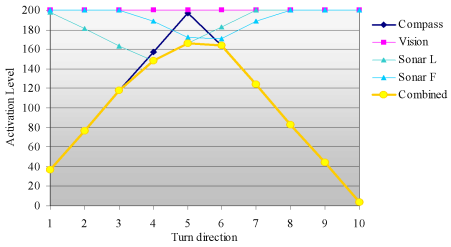
(b) Frame 2: Vision still in control



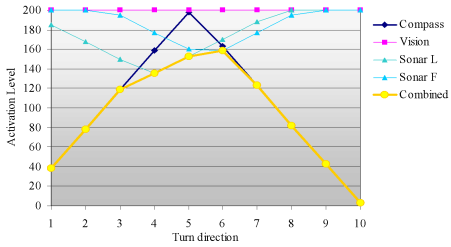
(c) Frame 3: Obstacle moving out of view



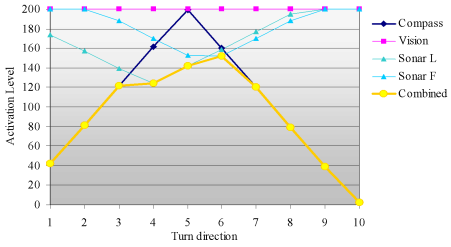
(d) Frame 4: Sonar sensors beginning to detect the obstacle



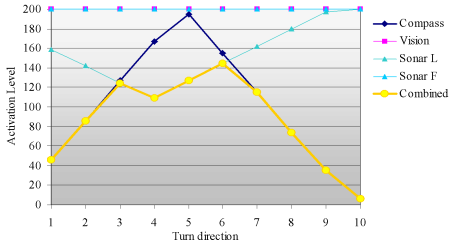
(e) Frame 5: Obstacle out of camera view; sonar takes over



(f) Frame 6: Obstacle draws closer



(g) Frame 7: Obstacle draws closer still



(h) Frame 8: Obstacle no longer detected by front sonar; left sonar ensures no collision

Figure 4.12: Interaction of Dynamic Fuzzy Sets during the first 8 frames of an obstacle avoidance experiment combining the stereo vision and sonar fuzzy control layers. In this experiment the obstacle was placed 100cm away.

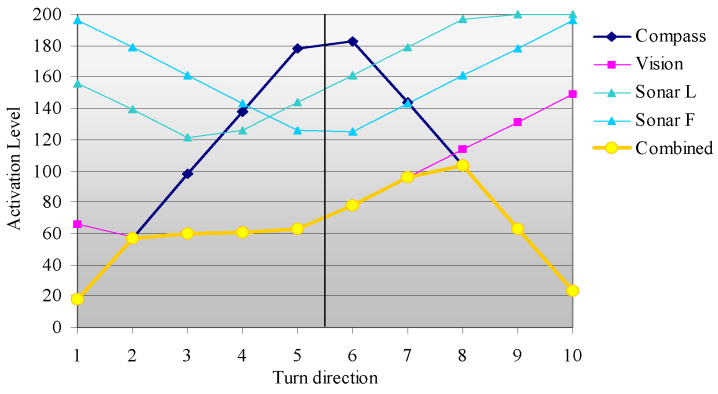


Figure 4.13: Interaction of Dynamic Fuzzy Sets during the first frame of the obstacle avoidance experiment combining the stereo vision and sonar fuzzy control layers. In this experiment the obstacle was placed 60cm away.

Chapter 5

Conclusion

The purpose of this study was to test whether combining stereoscopic vision and sonar using a simplified fuzzy logic control system achieves better obstacle avoidance than using stereo vision or sonar in isolation. "Better" was defined as being capable of responding to the presence of an obstacle as early as possible, and then proceeding to avoid a collision.

A stereoscopic vision system was developed based on colour image thresholding, edge based methods, and geometric stereo calculations. The error present in the system was tested and discussed. It was determined that the systematic error present in the readings were not significant given the nature the fuzzy control system being employed. Fuzzy control systems are concerned with dealing with uncertainty and vagueness-of-concept. Errors in sensor data, therefore, have only a minor impact on the final control decisions.

The control system employed was a new variation based on a fuzzy logic control system called a Dynamic Fuzzy Logic Control System . This technique was used to combine compass readings with sonar, stereo vision, and all three sensors together in order to test the hypothesis.

The tests conducted with the obstacle at an initial distance of 100cm yielded the following results. The vision layer was able to detect the obstacle from the first frame of the experiment and the sonar layers are able to avoid the obstacle when it was in close proximity to it. The combination of these layers enabled both of these benefits to be exploited—the combined approach turns to avoid the obstacle from the first frame of the experiment; considerably earlier than the sonar layers could. The combined approach succeeds in ultimately avoiding a collision where the vision system alone fails. Thus, the results of this study support the hypothesis.

It would of course be possible to use sensors with different characteristics, such as cameras with a wider field-of-view, or sonar sensors with characteristics

that allow for more fine-grained measurements. These improved sensors may indeed perform better for use in obstacle avoidance than those used in this study—better, perhaps, than the combined approach as well. Regardless of this, the fact remains that the approach used successfully combines the output of the sensors in question to achieve better obstacle avoidance compared with using them separately. This addresses the hypothesis being examined in this study.

The results from this study could be used as a starting point for additional research. Other sensor types could be tested in the context of the system. Sensors that have a particular unique advantage over other sensors would be particularly interesting to study. For example, infrared rangefinders have a generally have a narrower field of view and would therefore give more fine-grained results. A metal detector input could be used to avoid landmines. This could be combined with sensors able to detect obstacles such as sonar and/or stereo vision. The result would be a robot that is able to avoid landmines and obstacles whilst travelling through a minefield. Infrared cameras are able to detect heat sources that other sensors might not, especially when in the dark. Laser rangefinders are another popular sensor type that could be tried with the system.

The Dynamic Fuzzy Logic Control System Architecture described in this study could be built upon and extended to employ multiple output fuzzy conclusions. For example, rather than just making a decision about turn-direction, speed could be decided upon. The Centroid of Largest Area (CLA) defuzzification method could be considered as a replacement for the "winner takes all" defuzzification technique currently being employed. This should yield smoother transitions between behaviours.

In conclusion, the results found in this study support the hypothesis that combining short range sonar and stereoscopic vision using a simplified fuzzy logic control system is better than using these sensors in isolation in enabling the flexible robot platform to avoid obstacles. The errors present in the experiments were considered and it was determined that they do not have a significant impact on the results.

References

Acroname Inc. 2004, 'Devantech SRF04 Ranger', viewed 1 November 2006, <http://www.acroname.com/robotics/parts/R93-SRF04.html>.

Audrey Mbogho 2005, 'DirectShow Single-Frame Capture Class Without MFC', viewed 15 August 2006, <http://www.codeguru.com/cpp/g-m/directx/directshow/article.php/c9551/ >.

Brooks, R. 1986, 'A robust layered control system for a mobile robot', *Robotics and Automation, IEEE Journal of [legacy, pre-1988]* **2**(1), 14–23.

Brooks, R. 1991, 'Intelligence without representation', *Artificial Intelligence* **47**(1), 139–159.

Delgado, M., Gómez-Skarmeta, A. & Marín-Blázquez, J. 1998, 'Fuzzy Hybrid Techniques in Modeling', *Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems: Methodology and Tools in Knowledge-Based Systems* pp. 180–189.

Deriche, R., Zhang, Z., Luong, Q. & Faugeras, O. 1994, 'Robust recovery of the epipolar geometry for an uncalibrated stereo rig', *Proceedings of the third European conference on Computer vision (vol. 1) table of contents* pp. 567–576.

Devantech Ltd 2003*a*, 'SRF04—Ultra-Sonic Ranger Technical Specification', viewed 17 May 2006, <http://www.robot-electronics.co.uk/htm/srf04tech.htm>.

Devantech Ltd 2003*b*, 'Ultrasonic Rangers FAQ', viewed 17 May 2006, <http://www.robot-electronics.co.uk/htm/sonar_faq.htm>.

Devantech Ltd n.d.*a*, 'CMPS03—Compass Module FAQ', viewed 17 May 2006, <http://www.robot-electronics.co.uk/htm/cmpsqa.shtml>.

Devantech Ltd n.d.*b*, 'CMPS03—Robot Compass Module', viewed 17 May 2006, <http://www.robot-electronics.co.uk/htm/cmps3doc.shtml>.

Faugeras, O., Luong, Q. & Maybank, S. 1992, 'Camera Self-Calibration: Theory and Experiments', *Proceedings of the Second European Conference on Computer Vision* pp. 321–334.

Flanagan, C., Toal, D. & Leyden, M. 2003, 'Subsumption and Fuzzy-Logic, Experiments in Behavior-Based Control of Mobile Robots', *International Journal of Smart Engineering System Design* **5**(3), 161–175.

Flanagan, C., Toal, D. & Strunz, B. 1995, 'Subsumption Control of a Mobile Robot.', *Proceedings of the 16th Conference of Polymodel on Applications of Artificial Intelligence, Sunderland, UK* pp. 150–158.

Hall, D. J. 2005, 'Flexible robot platform for autonomous research'. Honours thesis, University of Tasmania.

Intel Corporation 2006, 'Open Source Computer Vision Library', viewed 26 July 2006, <http://www.intel.com/technology/computing/opencv/>.

Iocchi, L. & Konolige, K. 1998, 'A Multiresolution Stereo Vision System for Mobile Robots', *AIIA (Italian AI Association) Workshop, Padova, Italy* .

Langer, D. & Thorpe, C. E. 1997, *Intelligent Unmanned Ground Vehicles*, Kluwer Academic Publishers, Norwell, Massachusetts, chapter 9.

Leonard, J. & Durrant-Whyte, H. 1992, *Directed sonar sensing for mobile robot navigation*, Kluwer Academic Publishers, Cambridge, MA.

Lewis, H. W. 1997, *The foundations of fuzzy control*, Plenum Publishing Corporation, New York.

Leyden, M., Toal, D. & Flanagan, C. 1999, 'A Fuzzy Logic Based Navigation System for a mobile Robot', *Automatisierungs symposium Wismar* .

McComb, G. 2000, *Robot Builder's Bonanza*, McGraw-Hill/TAB Electronics.

Microsoft Corporation 2006, 'DirectShow', viewed 26 July 2006, <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/directshow/htm/directshow.asp>.

Mikeklein34 2004, 'Processing two Webcams', viewed 26 July 2006, <http://forum.java.sun.com/thread.jspa?forumID=28&threadID=510436 >.

Moravec, H. 1981, *Robot rover visual navigation*, UMI Research Press Ann Arbor, Mich.

Murphy, R. R. 2000, *Introduction to AI Robotics*, MIT Press, Cambridge, Massachusetts.

Murray, D. & Little, J. 2000, 'Using Real-Time Stereo Vision for Mobile Robot Navigation', *Autonomous Robots* **8**(2), 161–171.

Nalwa, V. S. 1993, *A guided tour of computer vision*, Addison-Wesley Publishing Company.

Nehmzow, U. 2003, *Mobile Robotics: A Practical Introduction*, Springer, London.

Payton, D., Rosenblatt, J. & Keirsey, D. 1990, 'Plan guided reaction', *IEEE Transactions on Systems, Man, and Cybernetics* **20**(6), 1370–1382.

Robert, L., Buffa, M. & Hebert, M. 1994, 'Weakly-Calibrated Stereo Perception for Rover Navigation', *Proc. Image Understanding Workshop* .

Rosenblatt, J. & Thorpe, C. 1995, 'Combining multiple goals in a behavior-based architecture', *Proc. IEEE Conference on Intelligent Robots and Systems* .

Sun Microsystems, Inc. 2006, 'Java Media Framework API (JMF)', viewed 26 July 2006, <http://java.sun.com/products/java-media/jmf/>.

Toal, D., Flanagan, C., Jones, C. & Strunz, B. 1996, 'Subsumption architecture for the control of robots', *IMC-13, Limerick* .

Tsai, R. 1987, 'A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses', *Robotics and Automation, IEEE Journal of [legacy, pre-1988]* **3**(4), 323–344.

Viswanathan, M. 2005, *Measurement Error And Research Design: A Practical Approach to the Intangibles of Research Design*, Sage Publications.

Yen, J. & Pfluger, N. 1995, 'A fuzzy logic based extension to Payton and Rosenblatt's command fusion method for mobile robot navigation', *Systems, Man and Cybernetics, IEEE Transactions on* **25**(6), 971–978.

Zadeh, L. 1965, 'Fuzzy Sets and Systems', *Information and Control* **8**(3), 338–353.

Zhang, Z. 1998, 'Determining the Epipolar Geometry and its Uncertainty: A Review', *International Journal of Computer Vision* **27**(2), 161–195.

Zhang, Z., Deriche, R., Faugeras, O. D. & Luong, Q.-T. 1995, 'A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry', *Artificial Intelligence* **78**(1-2), 87–119.

Zhang, Z. & Schenk, V. 1997, 'Self-maintaining camera calibration over time', *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on* pp. 231–236.

Appendix A

StereoAnalysis and StereoGrabber Code

The StereoAnalysis and StereoGrabber software for capturing and analysing stereo images written in C# is available on the CD-ROM.

Appendix B

Dynamic Fuzzy Logic Control System and OOPic Code

The Dynamic Fuzzy Logic Control System and OOPic Java code is available on the CD-ROM.

Appendix C

Tables of Measurements

Many measurements that were taken are available on the CD-ROM.

Appendix D

Aerial Images

The aerial images taken during all experiments are available on the CD-ROM.

Appendix E

Fuzzy Set Values

The outputs of the fuzzy sets used to make the control decisions during all experiments are available on the CD-ROM.